VALID MOTION ESTIMATION FOR SUPER-RESOLUTION IMAGE RECONSTRUCTION

A Dissertation
Presented to
The Academic Faculty

By

Michael Santoro

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in
Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
August 2012

VALID MOTION ESTIMATION FOR

SUPER-RESOLUTION IMAGE RECONSTRUCTION

Approved by:

Dr. Yucel Altunbasak
Advisor, Committee Chair
President
TÜBITAK

Dr. Ghassan Al-Regib
Co-advisor
Assoc. Professor, School of ECE
Georgia Institute of Technology

Dr. David V. Anderson
Assoc. Professor, School of ECE
Georgia Institute of Technology

Dr. Patricio A. Vela
Assoc. Professor, School of ECE
Georgia Institute of Technology

Dr. Brani Vidakovic
Professor, School of BME
Georgia Institute of Technology

Dr. Anthony J. Yezzi
Professor, School of ECE
Georgia Institute of Technology

Date Approved: May 2012

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

"Algún día en cualquier parte, en cualquier lugar indefectiblemente te encontrarás a ti mismo, y ésa, sólo ésa, puede ser la más feliz o la más amarga de tus horas."

–Pablo Neruda

ACKNOWLEDGMENT

First and foremost, I would like to thank my parents and family. Without them, my Ph.D. experience would not have been the same, and I consider myself lucky to have experienced as many things as I have. I would like to extend my thanks to my co-advisor, who saw my potential and provided much of his time and assistance over the last year of my thesis. To my committee of Dr. David Anderson, Dr. Anthony Yezzi, Dr. Patricio Vela, and Dr. Brani Vidakovic – thanks for your guidance and knowledge.

I have to thank several people at Georgia Tech who pushed and helped me along the path during difficult times – Gail Palmer, Dr. Russ Callen, Dr. Bonnie Ferri, Dr. Jim McClellan and Dr. David Hertling, among others. I would also like to thank several members from the Center for Signal and Image Processing (CSIP) for their friendship and knowledge. Thanks to (in no particular order) Mohsen Sardari, Arash Einolghozati, Nima Torabkhani, Ahmad Beirami, Aytac Azgin, Ibrahim Pekkucuksen, Osman Gokhan, Alireza Monfared, Brett Matthews, Salman Aslam, Amol Borkar, Salman Asif, Greg Krudysz, Mohammed Aabed, Dogancan Temel, Mingyu Chen, and any others I may have neglected to mention.

Special thanks goes out to my friends in Atlanta and abroad whom I have learned more from than any Ph.D. could provide me with. I would especially like to thank Anthon Sonnenberg, Maria Magdalena Carrasco, SJ, Stefan Grubic, Chester Thomas, Micah Butler, and Scott Morris. It has always been an adventure, and I expect many more to come. I also have to thank John Jameson and the many rewards programs that allowed me to see the world from a different perspective, whether temporarily or permanently.

CHAPTER 1

INTRODUCTION

The growing demand for high quality imaging has put a burden on hardware designers to develop better optics, better sensors, and more efficient designs. Unfortunately, we are now reaching fundamental limitations that no longer allow us to continue to shrink the size of transistors [4], and further progress is met with high cost and power limitations. Designers now face a demand for smaller devices as more people abandon large, high-quality cameras for handheld devices such as camera phones. These small, embedded cameras found in handheld devices cause significant image degradation as a result of poor optics and small sensor size. With small sensors, less light strikes the sensor elements, and therefore, it becomes difficult to maintain a high signal-to-noise ratio (SNR). Fortunately, we are able to overcome some of these hardware limitations through the use of digital image processing techniques, specifically, Super-Resolution (SR). SR has found use in applications such as surveillance video [5], remote sensing [6][7], medical imaging (CT, MRI, PET, Ultrasound, etc.)[8][9][10][11], and video standard conversion (e.g. NTSC to HDTV signal)[12].

SR aims to create a high resolution (HR) image from several low resolution (LR) images. The LR images are taken from the same scene by the image sensor, and under the right conditions, will contain non-redundant information. The goal of SR is to fuse the non-redundant information (pixels) into a single HR image. The HR image will, therefore, have a higher spatial resolution than any single LR image, where spatial resolution refers to the density of pixels per unit area. The higher the spatial resolution, the more details contained in the image. The amount of resolution increase, called the magnification factor, depends on the number of non-redundant LR images available from the scene [13]. However, as will be shown in this thesis, although SR can increase the spatial resolution, it may actually degrade the subjective quality of the HR image [14].

The subjective quality of the HR image suffers as a result of artifacts, which are generated during the fusion process as a result of erroneous motion vectors (MVs). Accurate motion estimation is the crux of the SR problem, and with erroneous MVs, SR may give worse results than interpolation methods [15]. Therefore, although it is necessary to provide accurate motion vectors in order to increase the spatial resolution, it is even more critical to be able to detect invalid motion vectors in order to prevent artifacts in the HR image.

The motion estimation problem is largely unsolved, and although much effort has been put forth to improve its accuracy, most motion estimation algorithms are too complex to be used in practical applications. Many practical applications require a real-time or online approach, whereas most algorithms in the literature take from several minutes to several hours to estimate the motion between two images. Therefore, we concentrate on low-complexity motion estimation approaches in this thesis. Low-complexity approaches generally require a block-based motion estimation algorithm and low-complexity priors. In addition, it is necessary that such an algorithm converges in a small number of iterations.

While a block-based motion estimation algorithm reduces the computational complexity, it is also presents its own set of challenges. Block-based parameters such as block size, search size, search strategy, and multiple minima all contribute to the difficulty of any block-based algorithm. In addition, a block-based algorithm must be able to accommodate for large motion between images without requiring an exhaustive search of the entire image. We look at how to optimize each of these parameters, and we demonstrate several improvements to make the block-based algorithm more robust.

Even with robust block-based parameters, it is generally not possible to produce accurate MVs with block matching alone. Small blocks are necessary to capture the independent motion of small objects and structures, but using small blocks increases the number of minima and contributes to the ill-posedness of the motion estimation problem. Therefore, it is necessary to regularize the motion estimation problem based on spatial or temporal data.

Toward this extent, spatial regularizers that enforce the "smoothness" of motion have been introduced. The smoothness of the motion field enforces the requirement that MVs in a local neighborhood should be similar. However, it is only possible to enforce the smoothness of MVs if it is known that the MVs belong to the same object. It is at this juncture that low-complexity and high-complexity motion estimation algorithms differ. To determine the optimal smoothness constraints, it is necessary to take into account the structure of the objects in the image to prevent the smoothness constraints from oversmoothing edges and object boundaries. However, such data-driven smoothness constraints are not suitable for low-complexity approaches. Therefore, we perform a sensitivity analysis to determine how to choose a low-complexity prior without oversmoothing the MVs at edges and object boundaries.

The downside of using low-complexity approaches is the increase in motion error. Therefore, when using low-complexity approaches, it becomes even more important to detect erroneous motion to prevent further application-related errors. In this thesis, we analyze several block-based approaches that have been introduced in the literature to characterize the validity of MVs. The main weakness of these methods is the dependence on neighboring MVs or manual thresholds. The similarity of a MV to its neighbors is not a good measure of validity if the neighboring MVs are invalid or are significantly different by construction. For many motions such as rotations and scalings, it is expected that there will be a significant deviation in the MVs of a local neighborhood. The second requirement of a manual threshold is typically a bad strategy for image processing applications where the image content can vary significantly from image to image. We show that such approaches will fail when tested using a wide variety of image sequences. To overcome the limitations of neighboring MVs and manual thresholds, we propose a block-overlap-based validity approach in this thesis. The block overlap validity approach is based on the overlap of motion-compensated blocks, which is directly related to the motion error.

After demonstrating that the block overlap validity approach is a good measure of MV

error, we examine its use as a regularizer. The main motivation for introducing an additional regularizer is to handle smooth regions, regions with brightness variations, and to reduce the dependence on search order. In many cases, the combination of block matching and smoothness constraints will produce multiple matches. In such cases, the chosen MV generally depends on the order in which MV candidates are tested in the smoothness constraints. However, we show that the block overlap regularizer helps to ameliorate this effect and improve the quality of the motion field.

In next chapters of this thesis, we first present the SR reconstruction model and identify the sources of error in Chapter 2.3 and Chapter 2.4. We show that preventing MV errors is critical to ensuring that artifacts do not appear in the HR image. In Chapter 2.5, we consider several different SR approaches in order to motivate our use of the projection-onto-convex-sets (POCS) method. Next, the motion estimation problem is introduced and several models are considered in Chapter 2.6. In this chapter, we also introduce the relevant background on block matching and regularization/smoothness constraints. In Chapter 3, we introduce the complete motion estimation framework as an energy minimization problem. Several improvements are made to the block matching and regularization terms in this chapter, and a novel method of using spatial MV priors is introduced. At the end of Chapter 3, a preliminary motion estimation algorithm is presented that combines the related background and improvements of previous chapters. Next, we develop the proposed validity metric in Chapter 4, and we show that the proposed method outperforms other validity methods in the literature. We also show two applications of the proposed validity method, de-interlacing and super-resolution. In both of these applications, the proposed validity metric is shown to improve the image quality. In Chapter 5, we demonstrate that the validity method of Chapter 4 can be adapted for use as an additional regularizer. The new regularizer is incorporated into the energy minimization framework, and in addition to reducing the MV error, provides reduced sensitivity to block size and a more uniform

distribution of motion-compensated blocks. The performance of the new energy minimization framework is shown to outperform several state-of-the-art methods in terms of motion vector error, interpolation error, and run time. Based on the success of our motion estimation algorithm, we evaluated its use in a new application – camera misalignment correction for depth estimation. In Chapter 6, we show that our motion estimation algorithm may be used to estimate angular, translational, and scaling misalignment introduced as a result of the mismatch between stereo cameras. Finally, we summarize our contributions and future research directions in Chapter 7. A complexity analysis for the complete SR algorithm is also given in Chapter 7, and the run time of the proposed motion estimation algorithm is compared to other state-of-the-art algorithms.

CHAPTER 2

BACKGROUND

In this chapter, we introduce the background for SR and motion estimation. First, we show the concept behind super-resolution using several low-resolution images. Next, we discuss the fundamental limitations of imaging systems and the sources of error that complicate the SR problem. After demonstrating several different SR approaches that have been used in the literature, we introduce the motion estimation problem and illustrate some of the difficulties the prevent us from obtaining error-free MVs. Next, the block-matching-based motion estimation framework introduced by Bierling [16] is discussed, and regularization/smoothness is introduced in terms of spatial MVs. At the end of this chapter, the stage is set for developing an energy minimization framework that combines both block matching and regularization in order to determine the optimal MV.

2.1 Super-Resolution Problem

While the SR problem can be formulated in multiple domains, it is most easily understood in the spatial domain. Therefore, before taking a look at the different SR approaches, we first introduce the basic idea behind SR in the spatial domain.

2.2 Super-Resolution Problem Formulation

To illustrate the basic idea behind SR, we assume that the image sensor produces four artifact-free LR images as shown in Fig. 2.1(a)-(d).

(a) Diagonals. (b) X pattern.

(c) Diamonds. (d) Circular arrows.

Figure 2.1. Artifact-free LR images.

We have exaggerated the spacing between sensor pixels and used poor resolution to illustrate the SR reconstruction process. The purpose of SR reconstruction is to fuse the LR images of Fig. 2.1 into one HR image. The images in Fig. 2.1 correspond to the patterns that are overlayed on the HR image shown in Fig. 2.2(a). The same HR image is shown without the patterns in Fig. 2.2(b). In this example, we have assumed that there is a pixel-wise displacement in both horizontal and vertical directions between each LR image. In practice, these displacements are sub-pixel displacements, and there are generally redundant pixels between LR images. In addition, more than four frames are usually needed to generate an HR image at twice the resolution of the LR images. However, the example serves to illustrate how the image resolution may be increased.

(a) Patterned HR image.

(b) HR image.

Figure 2.2. Reconstructed HR images.

2.3 Fundamental Limitations

To build a SR reconstruction model, we first examine the conditions necessary to capture the true scene, i.e., the ideal conditions. This analysis is important since the error in our model will come from our inability to satisfy these conditions.

To capture the true scene, the following conditions need to hold:

1. The image sensor has infinite spatial resolution.

2. The camera optics are free of aberrations and provide the correct focus, i.e., the system is diffraction-limited.

3. The camera shutter time is effectively zero.

4. No noise is introduced by the capture process.

In practice, these conditions are neither individually nor collectively satisfied. Since it is not possible to manufacture a sensor with infinite spatial resolution, 1) cannot hold. From optics theory, we know that it is not possible to eliminate all aberrations simultaneously. Providing the correct focus is a trade-off between aperture size and depth of field. In general, we cannot focus on every point in the scene simultaneously, which causes objects or different parts of objects to appear blurred. Therefore, 2) cannot hold. In addition, even with perfect optics, there is a fundamental maximum resolution given by the diffraction limit. If it were possible to make the camera shutter time effectively zero, this would result in almost no incident light on the image sensor, which would result in effectively zero SNR. Therefore, 3) cannot hold. Since the image sensor and camera components consists of electronics, there is always inherent noise in the system. Therefore, 4) cannot hold.

Each of these four conditions will contribute to the error in the HR image that we wish to reconstruct. Because of condition 1), the image captured by the sensor will contain aliasing due to finite spatial resolution. The aliasing manifests as blocky artifacts, which are more prominent around edges and textured regions. These effects are generally referred

to as sensor blur. For condition 2), we will collectively refer to the blur introduced by the aberrations and incorrect focus as out-of-focus blur. With non-zero camera shutter time in condition 3), the motion of objects in the scene or the motion of the camera during acquisition will cause motion blur. Finally, the noise induced by condition 4) will result in additive noise.

2.4 Super-Resolution Observation Model

With the above conditions in mind, we form a observation model that relates the original and HR image to the observed LR images. We are careful to distinguish the original image from the desired HR image. The HR image is reconstructed from the LR images, whereas the original image is the true scene that we wish to capture. Since it is not possible to reconstruct the original image, we must choose a magnification factor, L, for our desired HR image, where L = HR image resolution / LR image resolution. The value of the magnification factor will depend on the number of non-redundant LR images that are available. Following the observation model shown in Fig. 2.3, let $\mathbf{X}_c$ denote the continuous scene we wish to capture, and $\mathbf{X}$ be the desired HR image sampled above the Nyquist rate from the band-limited continuous scene. We represent the output of system, i.e., the k-th observed LR image from the image sensor, as $\mathbf{Y}_k$.

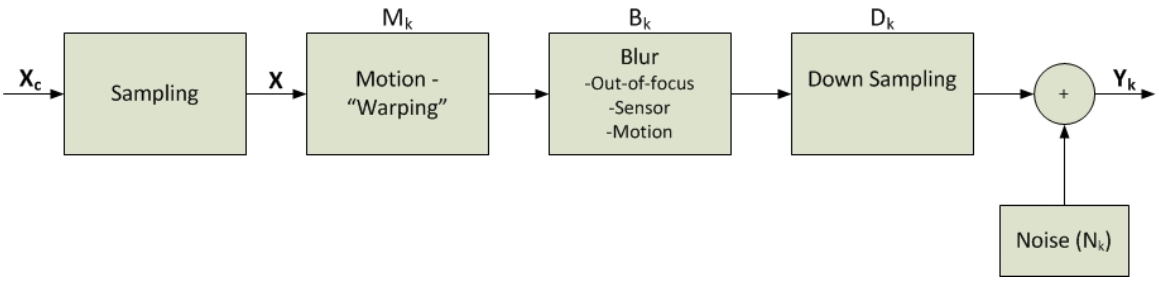


Figure 2.3. Super-Resolution Observation Model.

If we write $\mathbf{X}$ and $\mathbf{Y}_k$ in lexicographical order,

$$\mathbf{Y}_k = DB_kM_k\mathbf{X} + N_k, k = 1, 2, ..., K \tag{2.1}$$

relates the desired HR image $\mathbf{X}$ to the K LR images, $\mathbf{Y}_k$. In (2.1), D is a downsampling operator; B_k contains the blur for the k-th LR image; M_k contains the motion information that transforms the k-th LR image onto the HR image grid; and N_k is the noise in the k-th LR image.

The matrices D, B_k, M_k, and N_k are unknown and must be estimated from the LR images. We re-arrange the system into the form

$$\begin{bmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \\ \vdots \\ \mathbf{Y}_K \end{bmatrix} = \begin{bmatrix} DB_1M_1 \\ DB_2M_2 \\ \vdots \\ DB_KM_K \end{bmatrix} \mathbf{X} + \begin{bmatrix} N_1 \\ N_2 \\ \vdots \\ N_K \end{bmatrix}, \tag{2.2}$$

or equivalently

$$\mathbf{Y} = A\mathbf{X} + N, \tag{2.3}$$

where $A = DB_kM_k$. This is the Super-Resolution problem we wish to solve. If we apply linear system theory to (2.3), we can then write $\mathbf{X} = A^{-1}(\mathbf{Y} - N)$. However, the noise matrix N is unknown and matrix A is typically non-invertible and very sparse. It is necessary to constrain and regularize such an ill-posed system. We now examine Super-Resolution techniques proposed in the literature to estimate HR image $\mathbf{X}$ in (2.3).

2.5 Super-Resolution Approaches

It is widely regarded that Tsai and Huang were the first to show that Super-Resolution (SR) was theoretically possible [17]. Tsai and Huang formulated a frequency domain approach based on the shifting and aliasing properties of the continuous and discrete Fourier transforms. Although the idea was theoretically sound, others later found that the approach lacked the robustness needed to handle real images. Since Tsai and Huang's paper in 1984, multiple approaches have been published, and the problem has been formulated across multiple domains. However, SR still remains an unsolved and highly-researched problem. In

2008 alone, over 2000 papers were published on the topic [18].

Before we examine the different SR approaches, it is worth noting that all approaches suffer from similar problems. As noted by [18], each approach requires 1) subpixel motion estimation, 2) spatially-varying deblurring, and 3) robustness to error. It is difficult to satisfy any one of these requirements, let alone all three at the same time. Therefore, as we shall see in the following sections, each approach carries inherent limitations due to the inability to solve all three conditions simultaneously.

It is not our intention to introduce all of the approaches in the literature. For example, we do not discuss single frame, example-based approaches that require a large training database of image patches [19]. However, we wish to explore many of the commonly-used approaches to adequately characterize the breadth of the SR reconstruction problem and illustrate the inherent difficulties in producing a high-quality HR image.

2.5.1 Frequency Domain Approach

The frequency domain approach, first postulated by Tsai and Huang [17], is based on the shifting property of the Fourier transform and the aliasing relationship that exists between the Continuous Fourier Transform (CFT) of the original "true scene" and the Discrete Fourier Transform (DFT) of the LR images captured from the image sensor.

Let $\mathbf{x}(t_1, t_2)$ denote one continuous image of the true scene, and let $\mathbf{x}_k(t_1, t_2)$ denote the other $K-1$ continuous images of the true scene, each having a global vertical or horizontal displacement with respect to $\mathbf{x}(t_1, t_2)$. Therefore, we can write $\mathbf{x}_k(t_1, t_2) = \mathbf{x}(t_1+\Delta_{k_1}, t_2+\Delta_{k_2})$, where Δ_{k_1} and Δ_{k_2} are arbitrary but known displacements. Let us denote the CFT transform of $\mathbf{x}(t_1, t_2)$ as $\mathbf{X}(u, v)$ and the CFT of $\mathbf{x}_k(t_1, t_2)$ as $\mathbf{X}_k(u, v)$. Using the shifting property of the CFT, we can write the following:

$$\mathbf{X}_k(u, v) = \exp[j2\pi(\Delta_{k_1}u + \Delta_{k_2}v)]\mathbf{X}(u, v). \tag{2.4}$$

The shifted images are sampled with sampling period T_1 and T_2 to generate the LR image $y_k[n, m] = \mathbf{x}_k(nT_1 + \Delta_{k_1}, mT_2 + \Delta_{k_2})$. Let us denote the DFT of the LR images as $\Upsilon_k[\Omega_1, \Omega_2]$.

The DFT of the k-th LR image is related to the CFT by the aliasing property as follows:

$$\Upsilon_k[\Omega_1, \Omega_2] = \frac{1}{T_1 T_2} \sum_{l_1=-\infty}^{\infty} \sum_{l_2=-\infty}^{\infty} \mathbf{X}_k \left(\frac{2\pi}{T_1} \left(\frac{\Omega_1}{N_1} - l_1 \right), \frac{2\pi}{T_2} \left(\frac{\Omega_2}{N_2} - l_2 \right) \right), \tag{2.5}$$

where $N_1 \times N_2$ is the size of the LR image. Assuming that $\mathbf{X}(u, v)$ is band-limited, we can combine (2.4) and (2.5) to relate the DFT coefficients of $\Upsilon_k[\Omega_1, \Omega_2]$ to the CFT. Using matrix form, we write the following:

$$\boldsymbol{\Upsilon} = \boldsymbol{\Phi}\mathbf{X}, \tag{2.6}$$

where $\boldsymbol{\Upsilon}$ is a $K \times 1$ column vector whose k-th element represents $\Upsilon_k[\Omega_1, \Omega_2]$, $\mathbf{X}$ is a $N_1 N_2 \times 1$ column vector containing the unknown CFT coefficients, and $\boldsymbol{\Phi}$ relates $\boldsymbol{\Upsilon}$ and $\mathbf{X}$. Therefore, we wish to determine $\boldsymbol{\Phi}$ and solve for $\mathbf{X}$. We can then use the inverse DFT to obtain the reconstructed image.

The frequency domain approach assumes that the LR images are blur- and noise-free and global translations of each other. As discussed in Chapter 2.3, the LR images captured from the image sensor will contain both blur and noise. However, the assumption of a global translation between images is even more constraining. This assumption restricts independent motion of objects between LR images, i.e., all objects are constrained to move in the same direction. This assumption is almost always violated in real image sequences. Although attempts have been made to extend the frequency domain approach to handle blur and independent motions [20][21][22], these methods introduce complex models. Because of this, the majority of the methods in the literature focus on reconstruction in the spatial domain.

2.5.2 Nonuniform Interpolation Approaches

We now turn to spatial domain methods for SR reconstruction. To make the SR problem more tractable, we begin by making some assumptions and simplifications. Returning to the model in (2.1) from Chapter 2.4, we assume that B_k is the same for all K frames, i.e., $B_k = B$. We further simply our model by assuming that B is Linear and Spatial Invariant

(LSI), which allows us to write

$$\mathbf{Y}_k = DM_kB\mathbf{X} + N_k, k = 1, 2, ..., K. \tag{2.7}$$

We note that these assumptions are not valid for spatially varying blur such as out-of-focus blur, motion blur, etc. The model in (2.7) can be separated into the following three stages: 1) motion estimation between LR images, 2) nonuniform interpolation to align the LR pixels onto the HR grid, and 3) deblurring and noise removal to recover $\mathbf{X}$.

Several approaches have been proposed in the literature that perform a nonuniform interpolation of the LR image pixels followed by a deconvolutional algorithm with noise removal. Ur and Gross [23] generalized the multichannel sampling theorem of Papoulis [24] and Brown [25]. Alam et al. [26] proposed an interpolation scheme based on weighted nearest neighbors. Nguyen and Milanfar [27] proposed a wavelet-based interpolation algorithm. Danielyan et al. [28][29] proposed an algorithm based on adaptive filtering, and Hardie proposed an algorithm based on Weiner filtering [30].

Notable nonuniform interpolation approaches that have recently appeared in the SR literature are the nonlocal-means by Protter et al. [31] and the kernel regression methods of Takeda et al. [32]. In general, nonuniform methods are computationally efficient and allow for real-time applications. However, these approaches are limited in the simple models that they employ. The assumption that the blur and noise is the same for every LR image is not valid for real images. In addition, the reconstruction tends to be suboptimal since each stage is independent of the other stages. For example, errors in the motion estimation and interpolation stages are not taken into account in the final result. As we shall see later, the errors in the motion estimation step will have the greatest effect on the quality of the HR image.

2.5.3 Stochastic Approaches

In stochastic approaches, the HR image and motion between LR images are taken as random variables. Using the stochastic approach, we wish to maximize the probability of HR

image $\mathbf{X}$ given a vector-wise ordering of LR images, $\mathbf{Y}$, and the combined matrix A given in (2.3) from Chapter 2.4. If we assume that matrix A is already known, which is the usual (but limiting) assumption, we can express the maximization problem as follows:

$$\mathbf{X} = \arg\max_{\mathbf{X}} Pr(\mathbf{X}|\mathbf{Y}, A). \tag{2.8}$$

Using Bayes' rule, we recast the problem as

$$\mathbf{X} = \arg\max_{\mathbf{X}} Pr(\mathbf{Y}|\mathbf{X}, A)Pr(\mathbf{X}), \tag{2.9}$$

where $Pr(\mathbf{Y}|\mathbf{X}, A)$ is the data likelihood and $Pr(\mathbf{X})$ is the prior term for the desired HR image. Recalling (2.3) from Chapter 2.4, we express the noise component N as

$$N = \mathbf{Y} - A\mathbf{X}. \tag{2.10}$$

If we make the least-constraining assumption that N is Gaussian with zero-mean, the data likelihood can be expressed as follows:

$$Pr(\mathbf{Y}|\mathbf{X}, A) \propto \exp\left\{-\frac{1}{2\sigma^2}||\mathbf{Y} - A\mathbf{X}||^2\right\}. \tag{2.11}$$

The prior term, $Pr(\mathbf{X})$, can be expressed using many different distributions. Because of multiple possible sources of error (registration error, noise, blurring), we do not consider a uniform prior. Instead, we consider priors that regularize the ill-posed nature of the system. Comparisons of different image priors on the quality of the reconstructed HR image were evaluated in [33][19]. In the SR literature, $Pr(\mathbf{X})$ is typically given as an exponential Gibbs distribution:

$$Pr(\mathbf{X}) = \frac{1}{Z}\exp\{-\alpha V(\mathbf{X})\}, \tag{2.12}$$

where Z is a normalization factor, α weights the contribution of the prior, and $V(\mathbf{X})$ is a non-negative potential function. Given the data likelihood and HR image prior, we use (2.9) to form the Maximum a Posteriori (MAP) solution as follows:

$$\mathbf{X} = \arg\max_{\mathbf{X}} \frac{1}{Z}\exp\left\{-\frac{1}{2\sigma^2}||\mathbf{Y} - A\mathbf{X}||^2 - \alpha V(\mathbf{X})\right\}. \tag{2.13}$$

Minimizing the negative of the log-likelihood allows us to write

$$\mathbf{X} = \arg\min_{\mathbf{X}} \left\{ \frac{1}{2\sigma^2} \|\mathbf{Y} - A\mathbf{X}\|^2 + \alpha V(\mathbf{X}) \right\}, \tag{2.14}$$

where we have eliminated Z since it does not depend on $\mathbf{X}$. As mentioned in [18], many different priors have been proposed, but no single prior stands out as the best. However, the most commonly used priors are the Gaussian Markov Random Field (GMRF), the Huber MRF, and the Total Variation (TV) norm.

Regardless of the prior used, it is desired to regularize the HR image by choosing a prior that results in "natural images." Natural images tend to be highly non-Gaussian [34]; their distributions tend of have heavy tails. Therefore, use of the GRMF prior tends to over smooth the image by penalizing edges and textured regions. To improve upon the GRMF prior, the Huber MRF (HMRF) was proposed. The HRMF uses heavier tails than the GMRF and models the potential function, $V(\mathbf{X})$, as follows:

$$V(\mathbf{X}) = \left\{ \begin{array}{ll} \hat{\mathbf{X}}^2 & |\hat{\mathbf{X}}| \leq \gamma \\ 2\gamma|\hat{\mathbf{X}}| - \gamma^2 & \text{otherwise} \end{array} \right\}, \tag{2.15}$$

where $\hat{\mathbf{X}}$ is the first derivative of the image and γ is a threshold that allows the prior to both enforce smoothness and preserve edges [35]. The TV norm has received a lot of attention in the literature [36][37][38]. Like the HRMF, the TV norm preserves edges while enforcing smoothness. The TV prior can be expressed as follows:

$$V(\mathbf{X}) = \|\nabla\mathbf{X}\|_1, \tag{2.16}$$

where ∇ is a gradient operator that penalizes the total amount of change in the image and $\|\cdot\|_1$ represents the ℓ_1 norm. A robust version of TV, bilateral TV (BTV) was proposed in [39].

In the MAP approach discussed above, we have assumed that both the motion and blur are known. However, it has been shown that the HR image reconstruction can be improved if we treat the motion and blur as unknown parameters [40][41][42]. These approaches

are generally referred to as "Joint MAP Restoration." However, without simple parametric forms, these approaches are too complex to be used in real-world applications.

2.5.4 Projection onto Convex Sets Approach

The Projection onto Convex Sets (POCS) approach is a set theoretic restoration method. The POCS method incorporates prior knowledge about the solution into the reconstruction process by formulating multiple convex sets. Given that the sets have a nonempty intersection, the desired HR image lies within the sets. Note that even if the sets do not intersect, POCS will still find the minimum distance between the sets. To find the HR image, projection operators are used project from one set to the next given an initial HR estimate. Constraining the sets to be convex guarantees convergence of POCS. However, in the case that the intersection is empty, it is necessary to determine a stopping criterion to force convergence.

The choice of convex sets is very flexible; each set can represent different image priors and handle complex, nonparametric models. We introduce two sets that are directly applicable to SR reconstruction. The first set, commonly referred to as the data consistency constraint set, is based on (2.3) in Chapter 2.4. We define the data consistency constraint, one for each pixel in the LR images, as follows:

$$C_D[m_1, m_2, k] = \left\{ \mathbf{x}[n_1, n_2] : \left| r^{(\mathbf{x})}[m_1, m_2, k] \right| \leq \delta_0 \right\}, \tag{2.17}$$

where k is the current LR image in the set of K images, $\mathbf{x}[n_1, n_2]$ is the HR image pixel, δ_0 is the error we allow, and the error residual, $r^{(\mathbf{x})}[m_1, m_2, k]$, is defined as follows:

$$r^{(\mathbf{x})}[m_1, m_2, k] = \mathbf{y}[m_1, m_2, k] - \sum_{n_1, n_2} \mathbf{x}[n_1, n_2] h[m_1, m_2; n_1, n_2, k], \tag{2.18}$$

where $\mathbf{y}[m_1, m_2, k]$ is a LR pixel in the k-th frame, and $h[m_1, m_2; n_1, n_2, k]$ is the blur Point-Spread Function (PSF).

The second convex set applicable to SR reconstruction is the set of amplitude constraints that limits the amplitude of the pixel values. Given an eight-bit image, the amplitudes should be restricted from $A_1 = 0$ to $A_2 = 255$. We can express this set as follows:

$$C_A[m,n] = \{\mathbf{x}[m,n] \,|\, A_1 \leq \mathbf{x}[m,n] \leq A_2\}\,. \tag{2.19}$$

With the two convex sets given above, we are interested in finding the HR image, $\mathbf{X}$, which lies in the intersection of C_D and C_A, i.e., $C_D \cap C_A$. The POCS solution can be found using the following recursion:

$$\mathbf{X}_{k+1} = P_A P_D \mathbf{X}_k, \tag{2.20}$$

where $\mathbf{X}_0$ is an initial estimate of the HR image, and P_A, P_D are the projection operators for sets C_A, C_D, respectively. The initial estimate of the HR image, $\mathbf{X}_0$, is usually taken to be a interpolated version of one of the LR images.

POCS techniques have been proposed in the literature which handle space-varying PSFs, motion blur, sensor blur, arbitrary sampling lattices, non-finite aperture time, and non-zero aperture size [43][44][45][46]. The ability to incorporate these types of constraints and priors give POCS an advantage over stochastic approaches. However, a few issues that make POCS less than optimal are 1) non-unique solution, 2) slow convergence, and 3) assumed validity of motion parameters. Since the POCS solution depends on the initial estimate, the choice of $\mathbf{X}_0$ in (2.20) will determine the quality of the HR image [47]. Although POCS suffers from slow convergence, the convergence time can be improved by using relaxed projection operators [48]. The relaxed projection operator, T, can be expressed as

$$T = (1-\lambda)I + \lambda P;\ 0 < \lambda < 2, \tag{2.21}$$

where I is the identify matrix and P is the projection operator. The third issue, assumed validity of motion parameters, is generally the most constraining assumption. As previously mentioned, incorrect motion estimates will result in artifacts in the HR image. In fact, even a small number of incorrect estimates may render the HR image useless. To handle this,

previous methods have proposed validity maps that penalize image regions (and hence motion vectors) which tend to produce invalid motion [46][49]. However, this area is largely unexplored. We return to this topic in Chapter 4.

Since we wish to focus on the motion estimation problem and preventing errors that result from motion estimation, we chose the POCS SR method. Our choice was based on the fact that the POCS method is the most intuitive and least constraining SR method.

2.6 Motion Estimation Problem

All of the SR approaches introduced in the previous section require sub-pixel "true" motion estimation. However, the word "true" is actually a misnomer since the motion we wish to estimate is a projection of a 3-D scene onto the 2-D image plane. Therefore, the 2-D motion estimation problem is ill-posed by construction alone. In addition to the problem of estimating the motion of pixels whose true motion is in three dimensions; untextured regions, occlusions, deformations, and others types of complex motion further contribute to the ill-posedness of motion estimation. In this section, we look at the different models that may be used to estimate the motion, and we show that there are several types of motion for which there is no parametric model. After introducing the different models, we choose the local translation model because of its simplicity, and we introduce block matching and regularization using this model.

2.6.1 Motion Models

Several different models have been used to characterize the types of motion between two temporally adjacent images of the same scene. While no model can handle all motion types, examining the different models is helpful in understanding the sources of motion error. Toward this extent, we broadly categorize the motion models as a 1) global translation, 2) local translation, 3) parameter transformations, 4) complex transformations, and 5) occluded motion.

1. **Global Translation**. The simplest type of motion between two frames is a global

translation as shown in Fig. 2.4. In Fig. 2.4, we assume that there is only one object in the scene, namely, a hexagon. The pixels in the hexagonal shape move down and to the right as indicated by the arrows. A global translation assumes that every pixel $y_c[m, n]$ in the current image is shifted to $y_a[m + \Delta_m, n + \Delta_n]$ in the temporally adjacent image, where Δ_m and Δ_n are integers and $y_c[m, n]$, $y_a[m, n]$ denote pixels in the current and adjacent images, respectively. This is the same assumption made by the frequency domain approach of Chapter 2.5.1. While there are image sequences that satisfy this assumption (e.g. camera pan over a static scene), it is not valid in general.

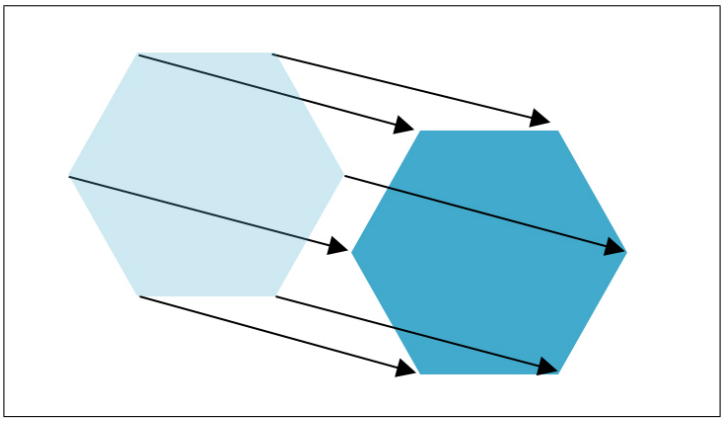


Figure 2.4. Global translation example.

2. **Local Translation**. Instead of assuming that every pixel in the current frame is a globally-shifted version of the pixel in the adjacent frame, the local translation model allows objects to move in different directions, as shown in Fig. 2.5. In Fig. 2.5, the pixels in the top hexagon move to the right, and the pixels in the bottom square move to the left.

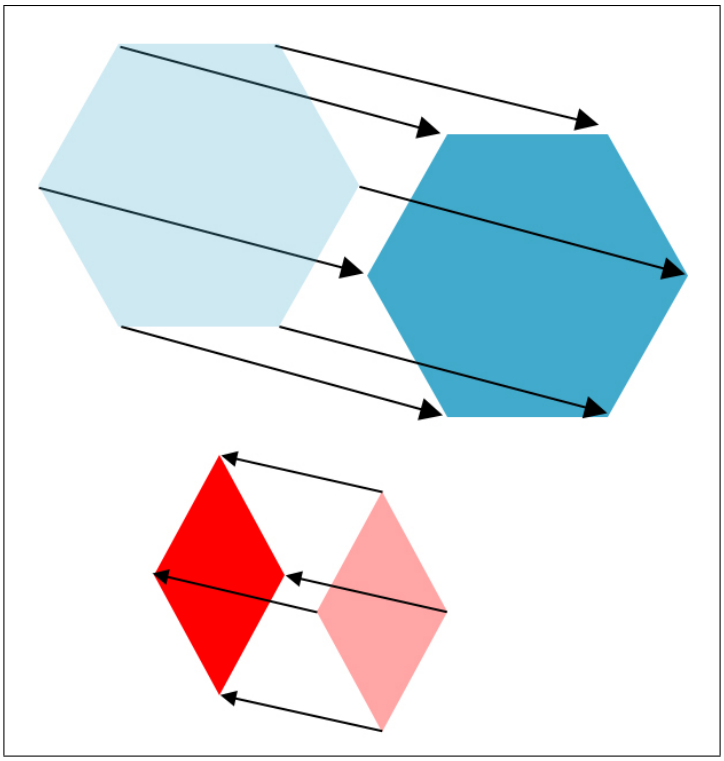


Figure 2.5. Local translation example

Within each region/object, the pixels are shifted by the same amount; i.e., given two matching regions (R_i, R_j) in the current frame and adjacent frame, respectively,

$$\sum_{m,n \in R_i} y_c[m,n] = \sum_{m,n \in R_j} y_a[m + \Delta_m, n + \Delta_n], \tag{2.22}$$

where Δ_m and Δ_n are fixed integers for a chosen pair (R_i, R_j) but may vary across different pairs. The local translation assumption allows for objects in the scene to have independent motions, i.e., objects are allowed to move in different directions. The major weakness of the local translation model is that it requires knowledge of the region boundaries. Since we do not know the segmentation of the objects in the image, it is possible to choose a region that is on a motion boundary. An example is shown in Fig. 2.6.

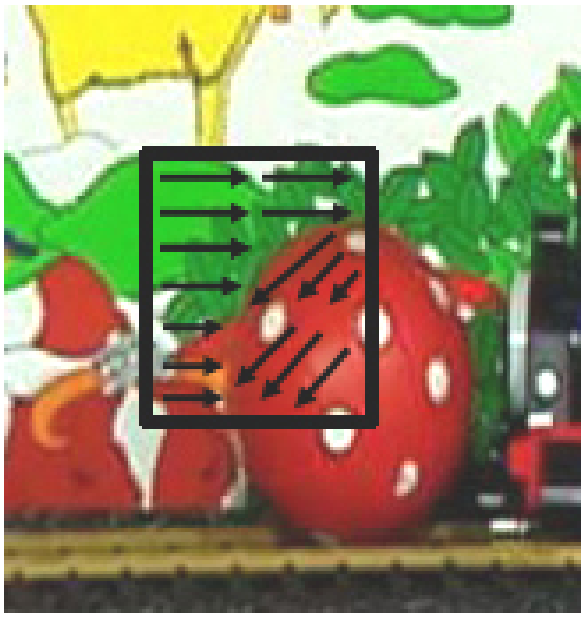


Figure 2.6. Example of local translation model failure.

In Fig. 2.6, the ball is rotating counter-clockwise and the background is moving to the right. We assume that the square represents the region of interest, and the direction of motion for each object is indicated by the arrows. Pixels within the square region indicated in Fig. 2.6 will undergo different motions. This violates the assumption that all pixels in the region should undergo the same translation. Another weakness of the local translational assumption can be described by a phenomenon known as the aperture problem. The aperture problem states that within a small window (aperture), different physical motions are indistinguishable. An example of this phenomenon is shown in Fig. 2.7.

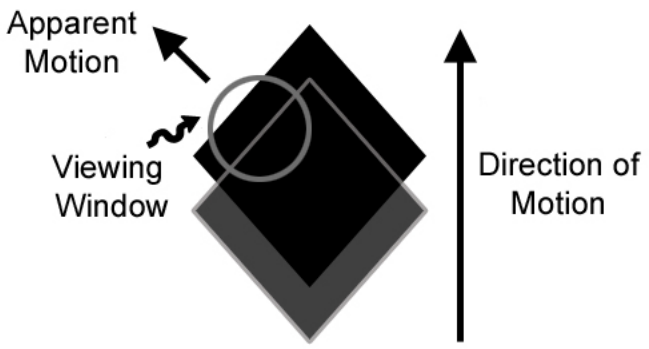


Figure 2.7. Example of aperture problem.

The true motion of the diamond in Fig. 2.7 is in the upward direction ("Direction of Motion"), as denoted by the arrow. However, if we confine our field of view to the inside of the circular viewing window, it will appear that the diagonal edge of the square is moving to the upper left (denoted by "Apparent Motion"). We can state the aperture problem more generally as follows: the motion of a homogeneous contour is locally ambiguous within an aperture that does not contain the entire contour.

3. **Parameter Transformations**. Other types of motion can be represented in mathematical form using parameter transformations such as the affine transformation, bilinear transformation, and perspective transformation. These types of transformation are considerably more complex than the translation models since they require several parameters to be estimated. For example, an affine transformation requires six parameters to be estimated, whereas a perspective transformation requires ten parameters to be estimated (assuming that object surface is planar). In addition, all of the estimated parameters require floating point precision, which limits the hardware-friendliness of such models. Since the affine transformation model is the most commonly used (to limit computational complexity), we only discuss this model. The affine model handles rotation, scaling, and/or shearing. For a pixel position $[m, n]$ in adjacent image

y_a, the pixel position $[m', n']$ in the current image y_c is given by the affine model as

$$\begin{bmatrix} m' \\ n' \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} a_0 \\ b_0 \end{bmatrix}, \tag{2.23}$$

where parameters a_0 and b_0 represent a translation, and parameters $\{a_1, a_2, b_1, b_2\}$ can be used to represent a rotation, scaling, and/or shearing.

4. **Complex Transformations**. We classify complex transformations as the deformation of an objects between temporally adjacent images such that the original form of the object has changed. One example of complex motion is shown in Fig. 2.8. In Fig. 2.8, the man's lips undergo different deformations that make it more difficult to estimate the motion.

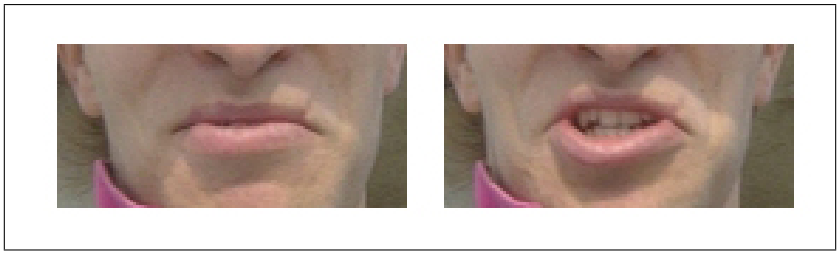


Figure 2.8. Example of complex motion.

The example shown in Fig. 2.8 is a complex motion because it requires a model that describes how the lips change from one image to the next. While scaling, shear, rotation, and translation can be handled by the affine motion model [50][51], the type of deformation shown in Fig. 2.8 generally requires a more complicated model. Node-based and mesh-based motion estimation have been proposed in the literature to handle deformation [52][53]. However, these methods are computationally expensive. Even more problematic, however, is that all of the models are sensitive to region selection. If a region is chosen such that it contains multiple objects with independent motions, all of these models will perform poorly.

5. **Occluded Motion**. Occlusion refers to the covering or uncovering of an object between temporally adjacent images. However, we broaden this category to include objects that appear in one image but not in the temporally adjacent image, e.g. disappearing objects or fast moving objects. An example of occlusion is shown in Fig. 2.9. In Fig. 2.9, the car that is moving to the left is occluded by the building.

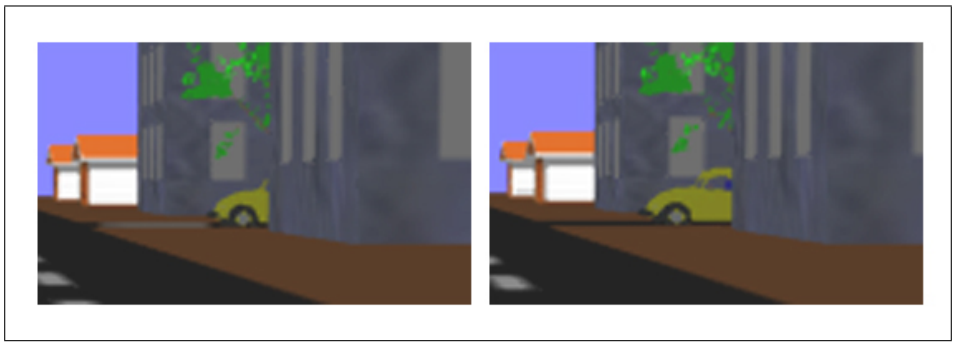


Figure 2.9. Example of occlusion.

For an occluded object, it is not possible to directly find a displacement vector which describes the motion of pixels between temporally adjacent images. In practical motion applications, it is generally sufficient to detect that an occlusion has occurred and mark the motion of the occluded object as invalid. In the context of SR, it is not possible to use the pixels where occluded motion has been detected since there is no pixel correspondence in the adjacent image.

Because of the model difficulties illustrated above, new methods have been proposed to avoid explicitly having to estimate the motion field [32][54]. These methods show promising results; however, they require pixel-level accurate MVs as an initialization [32].

Our intention is not to develop a solution that handles all possible motion types. To do so would result in a prohibitively expensive implementation, which is only the first step in the SR reconstruction. The majority of motion estimation algorithms in the literature use either block matching or optical flow to determine a dense motion field. Although

optical-flow-based algorithms provide superior motion vector (MV) quality over block-based algorithms [3], they do so at the expense of high computational complexity and long run times. In the interest of real-time applications, block matching algorithms provide a flexible trade-off between complexity and MV quality [55].

Block matching requires the use of the translational-motion model and brightness-constancy assumption to estimate the motion of blocks between image pairs. Unfortunately, these two requirements are often violated for real images; the actual motion can only be approximated as a translation for small displacements, and the brightness-constancy assumption does not hold for illumination changes due to non-uniform lighting, shadows, etc. Block matching is also sensitive to block size. Large blocks are needed to avoid local minima; however, large blocks produce poor matches compared to small blocks. To a large degree, the block size problem is minimized by using a hierarchical block matching framework (HBM) [16].

Even with the limitations of the translational-motion model and brightness-constancy assumptions, block matching algorithms perform surprisingly well. In the next section, we introduce the general block matching framework followed by the HBM framework.

2.6.2 General Block Matching

Block matching algorithms make use of the brightness constancy assumption, which assumes that image pixels retain their luminance values over a spatiotemporal displacement path, i.e.,

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t), \tag{2.24}$$

where $I(x, y, t)$ is a continuous representation of the pixel luminance; Δx and Δy represent the spatial shift; and Δt represents the temporal shift. The brightness constancy assumption in (2.24) is violated when the illumination of the scene changes between successive images; however, it is generally valid for small spatiotemporal displacements [56].

To make use of the brightness constancy assumption, block matching algorithms divide

the image into square regions generally referred to as blocks. To reduce complexity, the image is usually divided into blocks of *fixed* size; however, variable block size algorithms were proposed in [57][58]. Approaches that handle variable block sizes generally fall into the area of image segmentation, which is outside the scope of reduced-complexity block matching approaches. However, we note that different *square* block sizes will be useful in a hierarchical block matching framework, which we return to in Chapter 2.6.3.

With the image divided into blocks of predetermined size, the task of the block matching algorithm is to locate the block in the temporally adjacent image that best matches the block in the reference image. Note that the temporally adjacent image may fall before (backward block matching) or after (forward block matching) the reference image. In order to quantify the "best match," different metrics have been proposed [59] in accordance with (2.24). Correlation-based approaches are generally favored due to their robustness and low complexity [60]. Block matching methods in the literature almost exclusively use the Sum of Absolute Deviations (SAD) correlation metric. Given a block at position $[m, n]$ in the adjacent image, the SAD metric can be expressed as follows:

$$SAD[m, n] = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \Big| r[i, j] - s[m + i, n + j] \Big|, \tag{2.25}$$

where $N \times N$ is the block size, $r[i, j]$ is a pixel in the reference image, and $s[m + i, n + j]$ is a pixel in the temporally adjacent image.

The previous assumption of small spatiotemporal displacements between successive images in (2.24) implies that the block at position $[m, n]$ will be located in a neighborhood of the block at position $[i, j]$ in the reference frame. An example is shown in Fig. 2.10. In Fig. 2.10, the "concentric block" represents the block at position $[i, j]$ in the reference frame. A search window is formed around the concentric block in the adjacent frame to locate the block at position $[m, n]$ which minimizes the SAD metric.

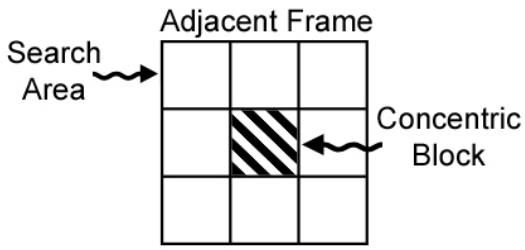


Figure 2.10. Forming search window around block to find minimum SAD.

The block (and hence motion vector(MV)) that minimizes the SAD will become the MV for the block at position $[i, j]$, i.e.,

$$MV_{i,j} = \{(k,l) \mid SAD(k,l) \leq SAD(m,n); -R \leq m,n \leq R-1\}, \tag{2.26}$$

where $MV_{i,j}$ is the MV for the block at position $[i, j]$ and $[-R, R-1]$ is the search range.

In order to maximize the probability of choosing the correct MV with the SAD metric, it is necessary to consider the following:

1. The choice of R for the search range.
2. The block size, B.
3. How to handle large motions between images.
4. Where to initialize the search.

To ease the sensitivity of 1), 2) and for large motions 3), we introduce the Hierarchical Block Matching (HBM) algorithm. To handle 4), we develop an initialization strategy in Chapter 3.2.

2.6.3 Motion Estimation via Hierarchical Block Matching

As discussed in the previous section, our ability to choose the correct MV depends on the block size and search range. The validity of the MVs is also influenced by block size. Choosing larger blocks will result in less false matches; however, a large block may

contain multiple objects with independent motions. Therefore, we desire an algorithm that combines the use of large and small blocks, where large blocks are used for an initial estimate of the motion, and smaller blocks are used to refine the estimate.

In order to accommodate for large displacements between images, it is necessary to increase the search range. However, it is prohibitively expensive to search the entire image for a matching block. Therefore, we wish to restrict the search range as much as possible in order to reduce computation time.

Hierarchical Block Matching (HBM) allows us to combine the advantages of small and large blocks, and the search range can be reduced as the algorithm progresses. The idea behind HBM is to create a pyramid for the pair of images whose motion we wish to estimate [16]. A three-level pyramid representation of two images is shown in Fig. 2.11. The top level in Fig. 2.11 is the lowest resolution image, which is obtained by low-pass filtering and subsampling the original image. The resolution of the images in the pyramid increases as the bottom level is reached. At the bottom level, the pyramid may contain the original image, or in the case of SR approaches, an interpolated version of the original image. The number of levels in the hierarchy will depend on the size of the original images as well as the desired sub-pixel accuracy for the MVs. Note that subpixel MVs may also be generated without interpolation [61][62][63][64].

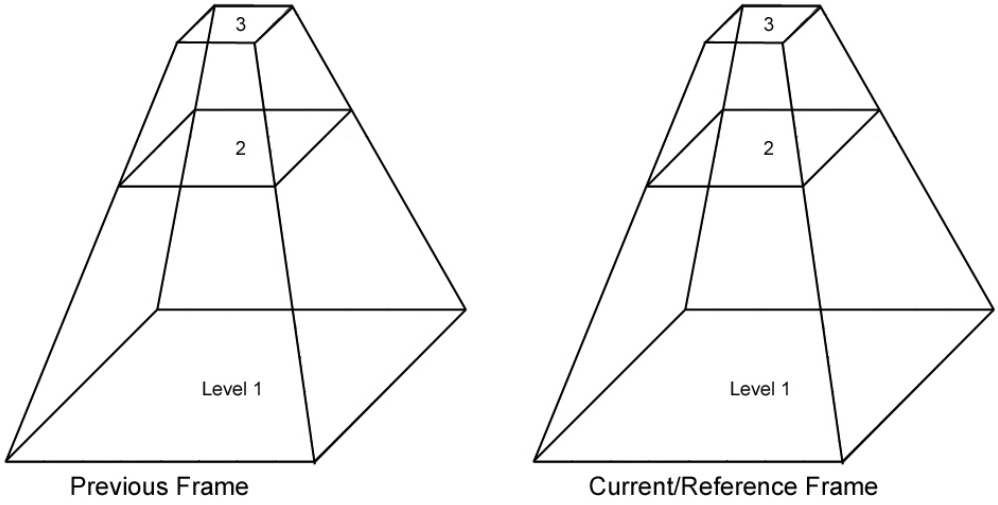


Figure 2.11. Hierarchical pyramid representation.

Following the creation of a pyramid for each image, the HBM algorithm performs block matching at each level successively, starting with the lowest resolution level [16]. The lowest resolution level uses large blocks and a modest search size to determine a rough estimate of the MVs. The MV estimates from the lower-resolution level are then passed up to the next higher-resolution level to initialize the search. As the algorithm progresses to a higher-resolution level, the search and block size may be decreased since the previous level provided an initial estimate. The progression of the HBM algorithm is illustrated in Fig. 2.12.

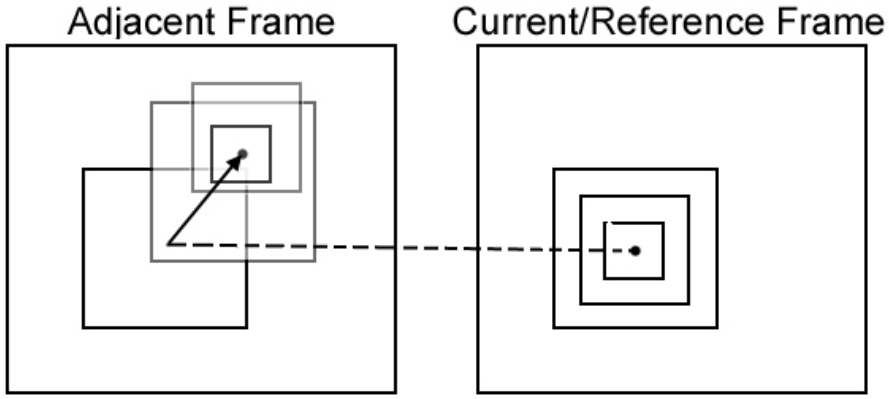


Figure 2.12. Hierarchical block matching progression.

Although HBM overcomes many of the difficulties associated with general block matching, it is important to note that the quality of matches at the next level of the hierarchy will depend on the quality of matches at the current level of the hierarchy; i.e., it is necessary to ensure that only valid motion vectors are passed to the next level of the hierarchy. Therefore, it is necessary to refine the MVs at each level before passing them to the next level of the hierarchy.

The SAD correlation metric introduced in Chapter 2.6.2 is a non-convex function; as the image resolution increases and the block size decreases in the HBM framework, the number of local minima increase. Therefore, it is necessary to introduce regularization in order to reduce the number of minima and make the motion estimation problem convex. In the next section, we introduce a regularization term that takes into account the spatial neighbors of the desired motion vector.

2.6.4 Regularization

To understand why it is necessary to regularize the results of block matching, consider the uniform region shown in Fig. 2.13. In Fig. 2.13, the block in the current/reference frame will match well with all of the blocks in the search window of the adjacent frame.

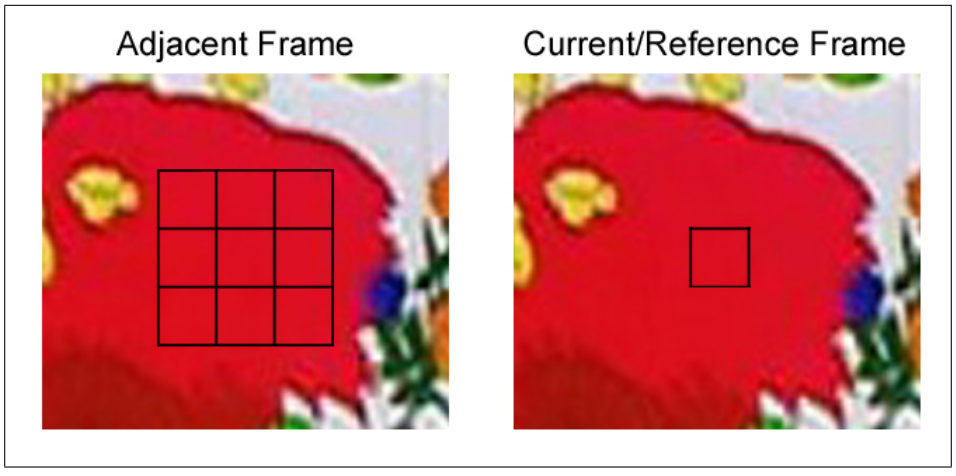


Figure 2.13. Example of multiple matches in a uniform region.

Therefore, without any prior knowledge or constraints, it is not possible to choose the

correct block. In real images, uniform regions and regions with repeating patterns are prevalent, making block matching an ill-posed problem.

Regularization methods introduce additional constraints in order to solve ill-posed problems. In the context of motion estimation, these constraints are generally referred to as smoothness constraints [65][66]. Smoothness constraints are based on the notion that blocks belonging to the same object should undergo the same displacement and thus have the same MV. As an example, consider the center block for the region shown in Fig. 2.14. The object in Fig. 2.14 is part of the calendar taken from the "Mobile and Calendar" sequence. We assume that the calendar is moving up and to the right, which is consistent with MVs of the neighboring blocks. The center block, however, shows a MV that is inconsistent with its neighbors.

Figure 2.14. Imposing smoothness using neighboring MVs.

If it is known that the center and neighboring blocks (and MVs) belong to the same object, smoothness requires that the center block have the same MV as its neighbors. Therefore, the MV of the center block should be replaced by one of its neighbors. Note that this is only half of the picture; the MV of the center block should only be replaced by one of its neighbors if it also minimizes the SAD. True motion estimation methods in the literature [65][67][68][69][70] have proposed a cost function which penalizes the deviation of the reference MV from its neighbors.

Before we evaluate different possible cost/penalty functions, it is first necessary to combine both block matching and regularization into a joint energy minimization framework.

CHAPTER 3

MOTION ESTIMATION FRAMEWORK

In the previous chapter, we chose the local translational model and introduced both block matching and regularization. In this chapter, we examine how to combine both block matching and regularization into a joint framework in order to determine the optimal MV. In this chapter, we are not interested in minimizing the block matching or regularization terms individually, but a joint minimization of both terms. We introduce a Lagrange multiplier to accelerate convergence of the MVs and to put more weight on the regularization term than the smoothness term. Following the development of the joint minimization framework, we discuss the steps necessary to develop a robust hierarchical-based motion estimation algorithm.

3.1 Energy Minimization Framework

Regardless of whether optical flow, block matching, or some other type of framework is used for motion estimation, the motion estimation problem can generally be formulated as the following energy minimization problem:

$$E = \min_i \{\mathcal{D}(I_0, I_1, v_i) + \lambda \mathcal{R}(v_i)\}, \tag{3.1}$$

where $\mathcal{D}(I_0, I_1, v_i)$ is a data term that measures the similarity of images I_0 and I_1 for a given MV v_i, $\mathcal{R}(v_i)$ is a regularization term which penalizes deviations in the smoothness of the motion field, and the Lagrange multiplier λ is used to weight the regularization term over the data term. The goal of the motion estimation problem is to choose a MV v_i such that the energy in (3.1) is minimized.

We now consider how to generate the energy minimization framework of (3.1) for block-based motion estimation. To do so, we develop a Bayesian framework that combines the SAD and smoothness constraints from Chapter 2.6.2 and Chapter 2.6.4 to determine

which MV minimizes the overall energy. It is important to note that we only consider spatial MVs in the smoothness constraints. For the first two frames a video sequence, there are no temporal MVs available, and therefore we can only consider the spatial MVs. To consider all spatial neighbors (e.g., horizontal, vertical, and diagonal), it is first necessary to determine a MV for each block in the image by minimizing with respect to the SAD only.

In terms of a Bayesian framework, we wish to maximize the probability of choosing a MV given the SAD error between motion-compensated blocks in the adjacent image and the spatial MVs of blocks in the current image. The MV v_i for the block under consideration in the current image and its spatial MVs v^s form a set of candidate MVs, $\mathbf{V}^{k\times k}$, where $k \times k$ is the size of the neighborhood. Using Bayes' theorem, we relate the current MV to the SAD error and spatial MVs as follows:

$$p(v_i \mid d, v^s) = \frac{p(d \mid v_i, v^s)p(v_i \mid v^s)}{p(d \mid v^s)}, \tag{3.2}$$

where d is the SAD error between the motion-compensated blocks, v^s contains the spatial MVs, and v_i is one of the MVs from $\mathbf{V}^{k\times k}$. We now examine each term on the right-hand side of (3.2). The first term, $p(d \mid v_i, v^s)$, can be written as $p(d \mid v_i)$ since the SAD error d only depends on the current MV and not its spatial neighbors. If we assume that the SAD error is additive, white, Gaussian noise, then $p(d \mid v_i)$ can be rewritten as

$$p(d \mid v_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2} \sum_{\mathbf{x}\in B} \left|I_1(\mathbf{x}) - I_0(\mathbf{x} + v_i)\right|\right\}, \tag{3.3}$$

where σ^2 is the variance of the pixel differences, $\mathbf{x}$ is the pixel position within a square block B of pixels, and I_1, I_0 represent the current and adjacent images, respectively. From (3.3), it can be seen that each pixel within block B of the current image $I_1(\mathbf{x})$ is subtracted from the corresponding motion-compensated pixel in the adjacent image, $I_0(\mathbf{x} + v_i)$.

The second term on the right-hand side of (3.2), $p(v_i \mid v^s)$, denotes the conditional probability of MV v_i given the spatial MVs, v^s. This term represents the prior term in the

Bayesian formulation, and under the assumption of having Markovian properties, can be expressed as a realization of a Gibbs random field [71]. We therefore express $p(v_i \mid v^s)$ as a Gibbs distribution as follows:

$$p(v_i \mid v^s) = \frac{1}{Z} \exp\{-U(v_i \mid v^s)\}, \tag{3.4}$$

where Z is a normalizing constant and $U(v_i \mid v^s)$ is an energy function which measures the similarity of MV v_i to the spatial MVs, v^s. We use the energy function to define the "smoothness" of the MV field. A MV field is described as smooth if the differences between the current MV and spatial MVs is small, and the energy is minimized. To characterize the smoothness, we wish to find a robust metric which penalizes the deviation of MVs. Therefore, we express the energy function of (3.4) as

$$U(v_i \mid v^s) = \sum_{j \in v^s} V(v_i, v_j), \tag{3.5}$$

where $V(v_i, v_j)$ is a function that assigns a penalty to the deviation of v_i and v_j.

The term in the denominator of (3.2) is not a function of v_i and can be replaced with a constant. Next, we combine (3.3), (3.4), and (3.5) to maximize the right-hand side of (3.2).

To find the MV $\hat{v_i}$ which maximizes the right-hand side of (3.2), i.e.,

$$\hat{v_i} = \arg\max_i p(d \mid v_i, v^s) p(v_i \mid v^s), \tag{3.6}$$

we substitute (3.3),(3.4), and (3.5). Therefore, (3.6) becomes

$$\hat{v_i} = \arg\max_i \frac{1}{Z\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2\sigma^2} \sum_{\mathbf{x} \in B} \left| I_1(\mathbf{x}) - I_0(\mathbf{x} + v_i) \right| - \sum_{j \in v^s} V(v_i, v_j) \right\}. \tag{3.7}$$

An equivalent representation of (3.7) can be formed by ignoring the constant terms and minimizing the negative logarithm, i.e.,

$$\hat{v_i} = \arg\min_i \left\{ \sum_{\mathbf{x} \in B} \left| I_1(\mathbf{x}) - I_0(\mathbf{x} + v_i) \right| + \sum_{j \in v^s} V(v_i, v_j) \right\}. \tag{3.8}$$

We re-write equation (3.1) for minimizing the overall energy below, which was introduced at the beginning of this section.

$$E = \min_i \{\mathcal{D}(I_0, I_1, v_i) + \lambda \mathcal{R}(v_i)\} \tag{3.9}$$

If we equate the terms in (3.9) and (3.8), we see that

$$\mathcal{D}(I_0, I_1, v_i) = \sum_{\mathbf{x}\in B} \left| I_1(\mathbf{x}) - I_0(\mathbf{x} + v_i) \right| \tag{3.10}$$

and

$$\mathcal{R}(v_i) = \sum_{j\in v^s} V(v_i, v_j). \tag{3.11}$$

Note that we have introduced Lagrange multiplier λ in (3.9) to weight the smoothness term over the data term. The choice of Lagrange multiplier is discussed in Chapter 3.3.

We now turn to the design of each term in (3.9). Minimization of the SAD term is discussed in the next section, and selection of the penalty function for the smoothness term is discussed in Chapter 3.3.

3.2 Block Matching Improvements

We expand upon the hierarchical block matching (HBM) algorithm proposed by Bierling [16] and discussed in Chapter 2.6.3. Specifically, we address the block search strategy used to minimize the SAD term in (3.10).

Block-matching-based algorithms in the literature form a search window in the adjacent frame around the block whose MV is to be determined as shown in Fig. 3.1. Then, a search for the block which minimizes the SAD is performed in raster scan order (indicated by arrows in Fig. 3.1).

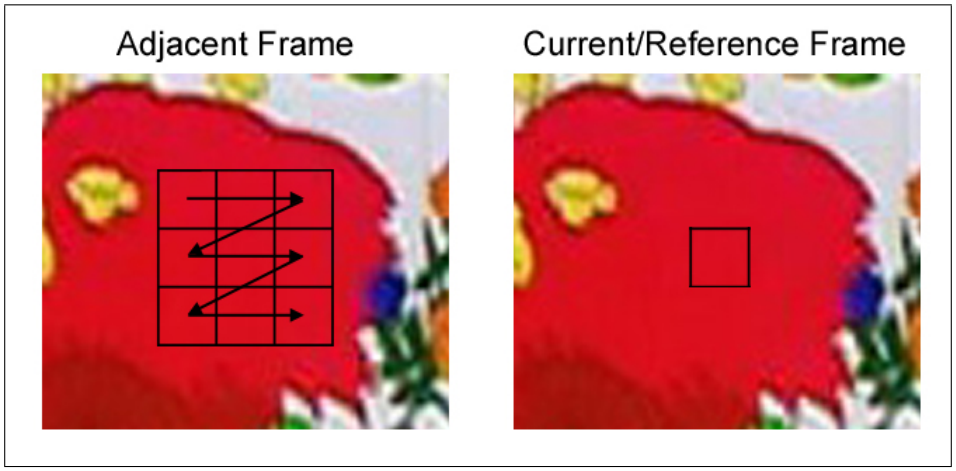


Figure 3.1. HBM using raster scan order.

To see why raster scan search is sub-optimal, consider a block that resides in a uniform region, i.e., the majority of the blocks in the search area have the same SAD error. In this case, the block in the top left corner of the search window will always be selected as the block with the minimum SAD value.

To improve the likelihood of selecting the best block in the event that multiple blocks produce the same SAD value, we propose a spiral search strategy. Spiral search relies on the observation that the block which minimizes the SAD error is likely to be in the vicinity of its corresponding block in the temporally adjacent frame. An example of spiral search is shown in Fig. 3.2.

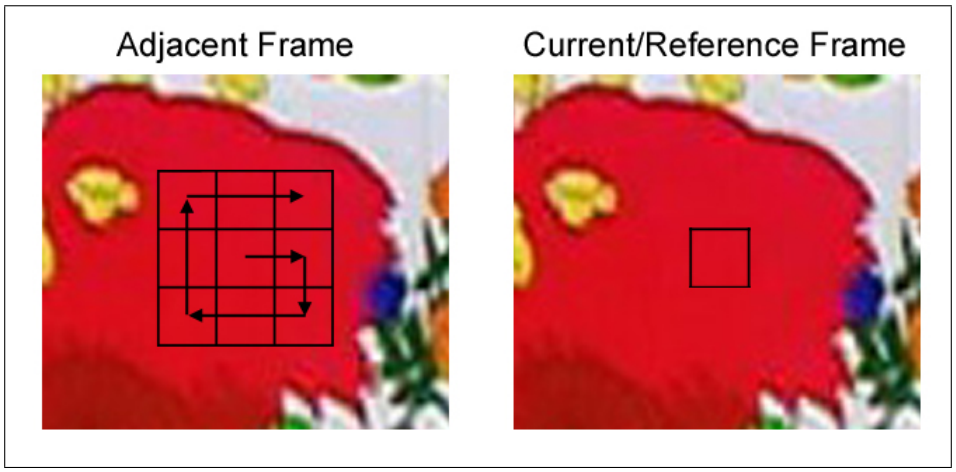


Figure 3.2. HBM using spiral search order.

As shown in Fig. 3.2, the search begins by comparing near neighbors before moving to blocks in the corners of the search window.

As previously discussed in Chapter 2.6.3, large displacements between temporally adjacent frames are handled by the HBM algorithm. Therefore, given an initial estimate of the motion, it is reasonable to assume that block which minimizes the SAD will be in the vicinity of the initial estimate. We show that spiral search performs better than raster scan in terms of the quality of MVs. To do so, real and synthetic image sequences from the Middlebury database [3] with ground truth MVs were used. These sequences cover a wide

array of image and motion types, and have become the standard benchmark in the literature. We chose sequences with nonrigid motion, realistic synthetic sequences, and stereo sequences of static scenes.

For the seven different image pairs listed in Table 1, a comparison of the endpoint error (EE) for raster scan and spiral search is shown. The EE is given as follows:

$$EE = \sqrt{(u - u_{GT})^2 + (v - v_{GT})^2}, \tag{3.12}$$

where (u, v) is the computed MV and (u_{GT}, v_{GT}) is the ground-truth MV. Note that the motion estimation algorithm used to produce the MVs for raster scan and spiral search contains both the data and regularization term in the energy framework of (3.9). However, since we delay the discussion of the regularization term until the next section, we do not give the details of the algorithm here. However, it suffices to say that the results for raster scan and spiral search both use the same motion estimation algorithm.

For the raster scan results in Table 1, the size of the block matching search area was reduced to approximately half of the size used for spiral search. Without this reduction, the EE for raster scan would be much larger.

Table 1. Improvement of spiral search over raster scan.

Image Pair	Raster Endpoint Error	Spiral Endpoint Error	Improv. in dB
Dimetrodon	0.292	0.254	0.60dB
Grove2	0.373	0.351	0.26dB
Hydrangea	0.332	0.276	0.80dB
Rubber	0.275	0.252	0.37dB
Urban2	2.75	0.572	6.82dB
Urban3	1.93	1.30	1.70dB
Venus	0.541	0.433	0.97dB

As shown in Table 1, spiral search outperforms raster scan for all of the sequences. The large improvement seen for the "Urban2" and "Urban3" sequences can be explained by the large motions present in those sequences. Raster scan is very sensitive to the size of the search area, and a large search area will result in many bad matches. However, a large search area is needed to capture large displacements that are present in the 'Urban2" and "Urban3" sequences. These conflicting requirements are greatly improved by spiral search.

3.3 Penalty Function Design & Lagrange Multiplier

We continue the development of the Bayesian framework from Chapter 3.1 by examining the second term of (3.9), namely, the regularization/smoothness term. For the smoothness term, it is necessary to determine how many MVs should be considered, i.e., the neighborhood size, as well as how to penalize deviations among MVs.

The two neighborhoods considered are referred to as first- and second-order neighborhoods, and are shown in Fig. 3.3. First-order neighborhoods contain the horizontal and vertical neighbors, and second-order neighborhoods contain the diagonal neighbors in addition to the horizontal and vertical neighbors. In Fig. 3.3, v_i represents the reference MV and the v^s's represent the spatial MVs.

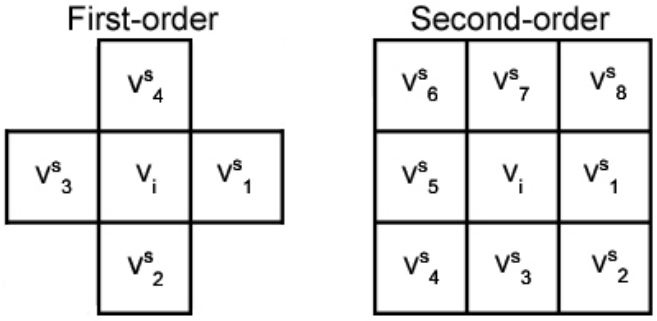


Figure 3.3. First- and second-order neighborhoods.

To penalize MV deviations, we consider the following penalty functions:

1. ℓ_1 norm: Smoothness $= \sum_{j \in v^s} \left\| v_i - v_j \right\|_1$. For MV v_i, each MV v_j in neighborhood v^s is

subtracted and summed.

2. ℓ_1 norm with global minimum: Smoothness $= \min_{j \in v^s} \left(\left\| v_i - v_j \right\|_1 \right)$. The MV v_j which has the minimum ℓ_1 norm with MV v_i determines the smoothness term.

3. ℓ_1 norm with median: Smoothness $= \| v_i - \text{median}\,(v_k) \|_1$, where $v_k \in \{v_i, v^s\}$ The ℓ_1 norm between v_i and the median MV of the set which includes v_i and neighbors v^s decides the smoothness term.

Using the spiral search of the previous section, we evaluate each of the three penalty functions for both first- and second-order neighborhoods. The image sequences used for comparison are taken from the real and synthetic sequences of [72] with known ground-truth MVs. The results for penalty functions 1), 2), and 3) are shown in Tables 2, 3, and 4, respectively. Note that all PSNR values are given in dB. Similar to the results for the raster scan versus spiral search of the previous section, we give the results for the different smoothness functions without having introduced the full motion estimation algorithm. It is sufficient to state that the penalty function is the only parameter that changes, and therefore the details of the algorithm can be left out. In all the results, the value of λ in (3.9) is set to one, i.e., the smoothness term is unweighted.

Table 2. PSNR (in dB) of motion vector error for L1 norm only.

Neighborhood	"Box"	"Checker"	"City"	"Office"	"Sphere"	"Street"
	PSNR	PSNR	PSNR	PSNR	PSNR	PSNR
First-order	36.47	26.71	38.03	43.77	46.41	47.89
Second-order	37.85	26.96	38.67	44.74	47.52	49.50

Table 3. PSNR (in dB) of motion vector error for L1 norm with global minimum.

Neighborhood	"Box"	"Checker"	"City"	"Office"	"Sphere"	"Street"
	PSNR	PSNR	PSNR	PSNR	PSNR	PSNR
First-order	35.62	26.11	37.73	43.17	45.28	47.55
Second-order	36.41	26.04	36.74	42.80	45.09	46.86

Table 4. PSNR (in dB) of motion vector error for L1 norm with median.

Neighborhood	"Box"	"Checker"	"City"	"Office"	"Sphere"	"Street"
	PSNR	PSNR	PSNR	PSNR	PSNR	PSNR
First-order	36.15	26.01	36.61	43.08	45.78	47.11
Second-order	35.46	26.02	37.69	42.73	45.11	47.51

From the results in Tables 2, 3, and 4, we conclude that the ℓ_1 norm produces the highest PSNR, a 1.79 dB average improvement over the next closest PSNR, the median penalty function. For the ℓ_1 norm, the choice of a second-order neighborhood over a first-order neighborhood results in a 3.35 dB average increase in PSNR. We also considered several other penalty functions such as a truncated linear model and averaging penalty function. However, we ruled out the truncated linear model because of the difficulty in choosing a truncation parameter. The averaging penalty function was ruled out because of the low PSNR compared to other penalty functions. Based on the superior performance and low-complexity of the ℓ_1 norm with the eight-connected neighborhood, we used this penalty

function and neighborhood in all subsequent chapters.

We now turn to the Lagrange multiplier λ given in (3.9). The purpose of the Lagrange multiplier is to weight the smoothness term over the data term. For regions with little or no texture, it is necessary to choose a large value of λ in order to propagate MVs from regions where there are not a large number of local minima, i.e., regions that contain more structure or texture. However, choosing a large value of λ will also oversmooth edges and motion boundaries. Therefore, to choose the optimal value of λ, the segmentation of the image must be known. However, in the interest of keeping the computational complexity low, we do not wish to find the segmentation of the image. Instead, we perform a sensitivity analysis to determine a small initial value for λ, and we increase the value of λ in proportion to the iteration number of (3.9). From our experiments, we have determined that approximately three iterations of (3.9) are necessary for the MVs to converge.

To conduct the sensitivity analysis, we used eight image sequences with known ground truth from the Middlebury [3] database. Since we have not yet discussed the details of the motion estimation algorithm, it suffices to say that all parameters in (3.9) were kept constant except for the value of λ. We varied the value of λ from 16 to 220, and for each of the three iterations, the value of λ was multiplied by the iteration number (e.g., $\lambda = 16$ in the first iteration, $\lambda = 32$ in the second iteration, and $\lambda = 48$ in the third iteration). The change in endpoint error for different values of lambda is shown in Fig. 3.4.

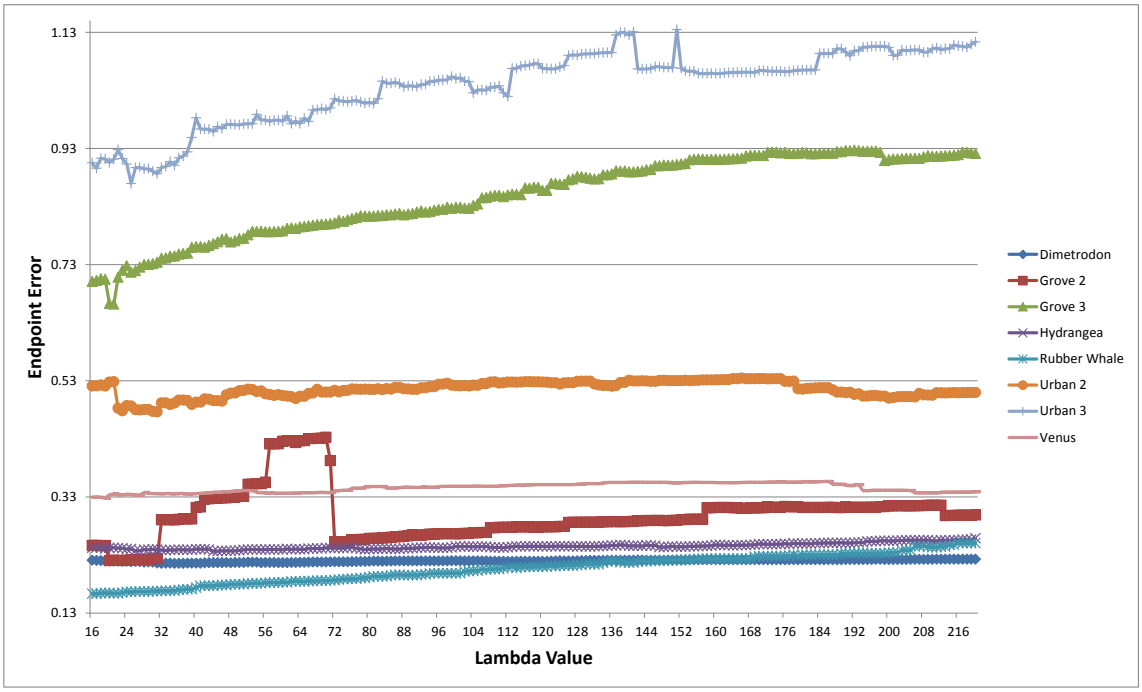


Figure 3.4. Comparison of endpoint error for different values of λ.

As shown in Fig. 3.4, the endpoint error is not very sensitive to the value of λ. Over the course of several experiments, we found that setting λ = block size provided the lowest endpoint error when averaged across a wide variety of image sequences. In Fig. 3.4, this would correspond to a λ value of 32.

There is a trade-off between choosing the value of λ and the design of the penalty function. For the "Grove 2," "Grove 3," and "Urban 3" sequences shown in Fig. 3.4, the endpoint error is shown to be more sensitive to the value of λ than for the other image sequences. The greater sensitivity is due to the large number of discontinuities present in these three sequences. When discontinuities are present, increasing the value of lambda will cause oversmoothing of the MVs on object boundaries. Therefore, it can be seen that in order to prevent oversmoothing, an image-driven penalty function is needed, i.e., it is necessary to locate discontinuities in the image and prevent the penalty function from assigning large values in these regions. Several image-driven penalty functions based on non-local total variation have been proposed in the literature [73][74]; however, the computational

complexity of such approaches makes them unsuitable for practical applications.

3.4 Adaptive Hierarchical Motion Estimation using Spatial Priors

In the previous section, we discussed first- and second-order spatial MVs with regard to the current level in the image hierarchy. However, when utilizing a hierarchical block matching (HBM) framework, the size of the candidate set may be increased after the MVs for the first (lowest resolution) level of the hierarchy have been determined. The additional MVs in the candidate set are taken from the spatial MVs at the previous level of the hierarchy. If an eight-connected neighborhood is used in the smoothness constraints, the expanded candidate set will consist of 18 spatial MVs for the desired image pair. The expanded candidate set is shown in Fig. 3.5.

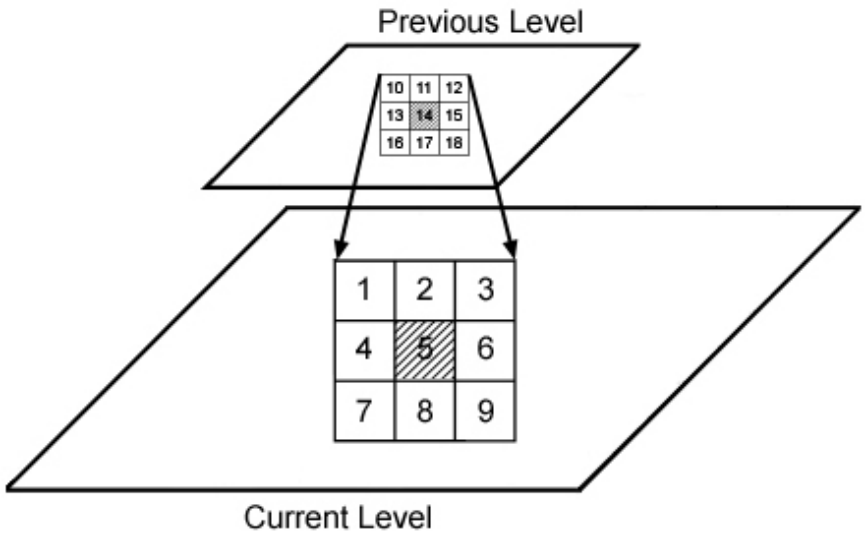


Figure 3.5. Candidate set using two levels of hierarchy.

As shown in Fig. 3.5, MV '5' (shaded) is the reference MV for the current level, and MV '14' (shaded) is the corresponding MV for the previous level of the hierarchy. To determine the MV that minimizes the energy, each of the possible 18 MVs should be tested in (3.9). We refer to this method as multiple candidate search (MCS), and it will serve as a basis for comparison with the proposed method.

To motivate the proposed method, we examine two possible cases where the block matching search will fail. For the spiral search strategy introduced in Chapter 3.2, the initial search direction is ambiguous. Rather than performing the search in the clockwise direction, the search could also be performed in the counter-clockwise direction. In addition, the first searched block could be any of the neighboring blocks.

Two cases where multiple matches may exist depending on the search direction are shown in Fig. 3.6.

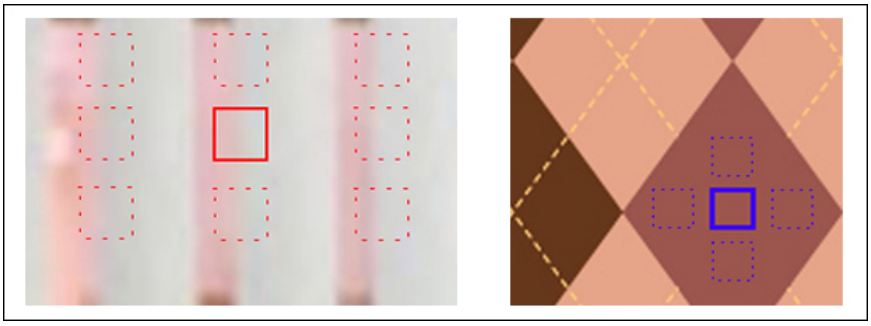


Figure 3.6. Examples of images containing multiple matches.

The image on the left contains vertical window blinds that repeat in the horizontal direction. The solid block in the image represents the block whose MV we wish to determine, and the blocks with dotted lines represent possible matches. The image on the right contains a pattern taken from a textured region. Similarly, the solid block represents the block whose MV we wish to determine, and the blocks with the dotted lines represent possible matches.

Even in the absence of motion, there will be multiple minimums for the blocks in both images. However, a unique minimum can be found in both images if a larger block size is used. Fortunately, the HBM framework is well-suited to handle such cases. In the HBM framework, the initial level of the hierarchy contains large blocks which provide an initial estimate of the motion. Therefore, we wish to take advantage of the previous level's MV to infer the best matching block at the current level of the hierarchy.

To solve for the problem of multiple matches in Fig. 3.6, other works [66][70] have

introduced new MVs into the candidate set by adding normal distributed noise, i.e.,

$$\mathbf{v}_{new} = \left\{ \mathbf{v}_{old} + \mathbf{n} \mid \mathbf{n} \sim N(0, \sigma^2) \right\}, \tag{3.13}$$

where $\mathbf{v}_{old}$ is one of the MVs in the original candidate set, and $\mathbf{v}_{new}$ is a new MV introduced into the candidate set by adding normal distributed noise, $\mathbf{n}$. However, we do not consider this approach for various reasons: 1) it is difficult to determine how many candidates to include and how to choose the value of σ; 2) since new candidates are randomly introduced without regard to the data, it is possible that a bad minimum may be introduced in the candidate set; 3) the computation time significantly increases as more candidates must be tested in (3.9).

In the proposed method, we introduce two energy terms similar to (3.9). The first term, SAD_{min}, represents the minimum SAD value for the current level of the hierarchy without regard to any spatial MVs from the previous level. The second term, Smoothness_{min}, represents the MV that has the smallest penalty with the previous level's MVs. This term is computed for each searched block. We then form the following two expressions:

$$\begin{aligned} E_1 &= \text{SAD}_{min} + \text{Smoothness}_1 \\ E_2 &= \text{SAD}_2 \quad + \text{Smoothness}_{min}, \end{aligned} \tag{3.14}$$

where Smoothness_1 is the penalty (using previous level's MVs) for the MV determined by SAD_{min}, and SAD_2 is the SAD value for the block whose MV produced the Smoothness_{min} value.

The decision rule for choosing one of the two possible MVs is given as follows:

$$\begin{aligned} &if\ (E_2 < E_1) \\ &\quad \text{choose } MV_2 \\ &else \\ &\quad \text{choose } MV_1, \end{aligned} \tag{3.15}$$

where MV_1 is the MV corresponding to E_1 and MV_2 is the MV corresponding to E_2. The decision rule in (3.15) is based on empirical evidence which suggests that greater preference should be given to the block which minimizes the SAD error (E_1) rather than the block which minimizes the MV penalty, i.e., E_2.

3.4.1 Results

In this section, we show that using smoothness constraints in HBM improves the quality of the motion field. All of the results shown in this section were generated using the Middlebury test sequences with known ground-truth MVs [3].

We compare the proposed method of introducing smoothness constraints into HBM with MCS using the endpoint error metric, which is given as follows:

$$EE = \sqrt{(u - u_{GT})^2 + (v - v_{GT})^2}. \tag{3.16}$$

In (3.16), (u, v) is the computed MV and (u_{GT}, v_{GT}) is the ground-truth MV.

Image Pair	MCS Endpoint Error	Proposed Endpoint Error	Improv. in dB
Grove2	0.353	0.330	0.30dB
Grove3	0.813	0.793	0.11dB
Hydrangea	0.277	0.270	0.11dB
Rubber	0.252	0.245	0.12dB
Urban2	0.579	0.565	0.11dB
Urban3	1.32	1.21	0.38dB
Venus	0.434	0.391	0.45dB

Table 5. Improvement of proposed algorithm over MCS.

As shown in Table 5, the proposed algorithm results in an improvement for all of the test sequences. The 'Venus" sequence showed the largest improvement (0.45dB), and the average improvement for all sequences was 0.23dB.

The combination of spiral search and smoothness constraints in HBM also reduces the required number of bits needed to represent the MVs. In a video coding context, this reduces the number of bits to be transmitted. For the sequences shown in Table 5, the proposed methods save an average of 1632 bits (204 bytes) per image pair.

3.4.2 Summary

The results in this section show that applying smoothness constraints in HBM produces an improvement in the quality of the motion field without increasing the size of the candidate set or introducing bad minimums. For the "Grove2", "Urban3", and "Venus" sequences of Table 5, which contain large motion discontinuities, the proposed method was shown to significantly outperform the MCS approach.

3.5 Preliminary Motion Estimation Algorithm

In this section, we combine the results of the previous sections in order to present a preliminary motion estimation algorithm. We consider how many levels the image pyramid should have in an HBM framework, how to choose the block size and search size, and how to refine the MVs at each level of the hierarchy before passing the MVs to the next level of the hierarchy. The results given in this section are based on experiments using several image and video sequences, and have been shown to be robust for different image sizes and frame rates. Improvements to the preliminary algorithm introduced in this section will be made in subsequent chapters.

The original energy minimization problem has not changed; we still wish to find the MV which minimizes the overall energy. The energy minimization equation is re-written as

$$E = \min_i \{\mathcal{D}(I_0, I_1, v_i) + \lambda \mathcal{R}(v_i)\}. \tag{3.17}$$

We first consider improvements related to the data term, $\mathcal{D}(I_0, I_1, v_i)$. In Chapter 3.2, the hierarchical block matching framework was introduced, and it was shown that the spiral

search order is an improvement over raster scan. Next, improvements related to the regularization term $\mathcal{R}(v_i)$ were considered. In Chapter 2.6.4 and Chapter 3.3, we discussed regularization and smoothness constraints, and several different penalty functions were analyzed. It was found that the ℓ_1 norm provided the best MV quality in terms of endpoint error. In Chapter 3.3, the sensitivity of endpoint error to the value of λ was discussed, and it was found that setting λ = block size provided the best trade-off in minimizing the endpoint error. We now look at how each of these individual parts fits into the motion estimation algorithm.

To begin, it is necessary to determine both the image resolution and the number of levels in the image pyramid that will be used for hierarchical block matching. Using more levels in the pyramid saves computation time, but as shown in [75], it may also degrade the quality of the MVs. Generally, using a pyramid with three levels provides a good trade-off between MV quality and performance [75]. To determine the size of the lowest resolution image in the pyramid, it is first necessary to determine the accuracy of MVs desired. Since we will later be interested in using SR to produce HR images with a 2X magnification in spatial resolution, the highest resolution level of the pyramid should contain an image interpolated by a factor of two[1]. Therefore, the lowest resolution level of the pyramid will be half the size of the original image. To generate the image at the lowest resolution level, it is first necessary to perform Gaussian smoothing prior to downsampling the image in order to prevent aliasing and provide denoising. A three-level pyramid constructed in this manner will provide half-pixel accurate MVs.

The next issue to address is the size of blocks and search areas for each level of the pyramid. Using a wide range of image sequences, our research has shown that the following block and search sizes are generally sufficient to handle different image sizes and large displacements:

[1] Subpixel MVs may also be generated without interpolation [61][62][63][64].

Table 6. Block sizes and search sizes for three-level image pyramid

Level	Block Size	Search Size
Level 1	8	32
Level 2	16	64
Level 3	32	128

Note that "Level 1" represents the highest resolution level of the pyramid, and "Level 3" represents the lowest resolution level of the pyramid. In addition to handling different image sizes and displacements, the block sizes in Table 6 also minimize the number of local minima that are generated as a result of block matching. We found that using a block size smaller than 8x8 substantially increases the number of minima, which conversely affects the overall quality of the motion field. While it is fairly easy to generate image sequences for which the parameters of Table 6 will not perform well, these parameters perform well for images/videos having QCIF, CIF, and VGA resolutions and a frame rate of at least 15 frames-per-second.

We now consider energy minimization for one level of the image pyramid. Energy minimization begins at the lowest resolution level of the pyramid, and the lowest-resolution level provides an initial estimate for initializing the block matching search at the next higher resolution level of the pyramid.

Since the regularization term $\mathcal{R}(v_i)$ in (3.17) requires eight spatial MVs, it is first necessary to determine a MV for each image block prior to minimizing the overall energy of (3.17). The initial MVs are determined by minimizing the energy with respect to $\mathcal{D}(I_0, I_1, v_i)$ only, i.e., minimizing with respect to the SAD metric.

With the initial MVs from SAD minimization, the overall energy can be minimized by applying (3.17). Given the initial MV estimate and the eight spatial MVs, there will be a total of nine MV candidates that must be tested in (3.17). The candidate that results in

the smallest energy will be chosen as the optimal MV. In the event that more than one MV produces the same energy, the chosen MV will depend on the order in which the candidates are tested in (3.17). Although it has been suggested to keep multiple MVs for each block in such cases, we did not notice a significant improvement in the quality of the motion field to justify the increase in computation time. In Chapter 5, we introduce an additional regularizer to ameliorate the effect of multiple MVs that produce the same energy. As discussed in Chapter 3.3, we perform three iterations of (3.17) in order for the MVs to converge, and the value of λ is increased in proportion to the iteration number.

After the optimal MV has been chosen for the initial block size (32x32 for the lowest resolution level), it is necessary to reduce the block size and refine the MVs using smaller blocks. Starting with the initial block size of 32x32, the block size is halved (16x16), and the MVs for the halved blocks take on the MVs of the parent block, as shown in Fig. 3.7.

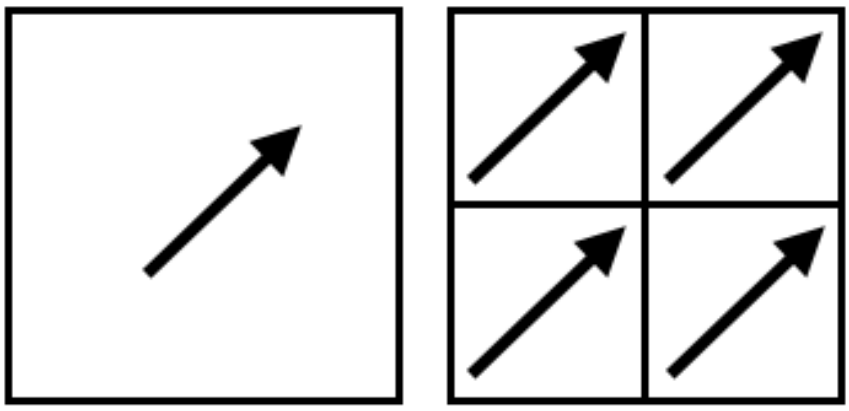


Figure 3.7. Splitting MVs for reduced block sizes.

In Fig. 3.7, the block on the left represents the 32x32 block size, and the blocks on the right are the individual 16x16 blocks.

Although it is only necessary to reduce the block size to half of the block sized used at the next level (e.g., block size at Level 3 should be reduce to at least 8x8 to cover 16x16 blocks at Level 2), reducing the block size down to 2x2 blocks produces MVs of higher quality. Each time the block size is reduced, the energy is minimized with respect to the

new MVs using three iterations of (3.17), which was discussed in Chapter 3.3.

We are now ready to put together the complete preliminary algorithm. The algorithm (Algorithm 1) is given in Fig. 3.8 and illustrated in Fig. 3.9.

Algorithm 1 : Preliminary Motion Estimation Algorithm

1: Form image hierarchy and begin at lowest-resolution level.
2: For all blocks in image I_1, find the corresponding blocks in image I_0 with the lowest SAD error.
3: Using the MVs from Line 2, find the MV with the minimum energy.
4: Iterate Line 3 until MVs converge, reduce the block size, and repeat until block size is 1x1.
5: Pass converged MVs to next level of hierarchy and return to Line 2. Repeat until highest-resolution level of hierarchy is reached.

Figure 3.8. Algorithm 1 - Preliminary Motion Estimation Algorithm

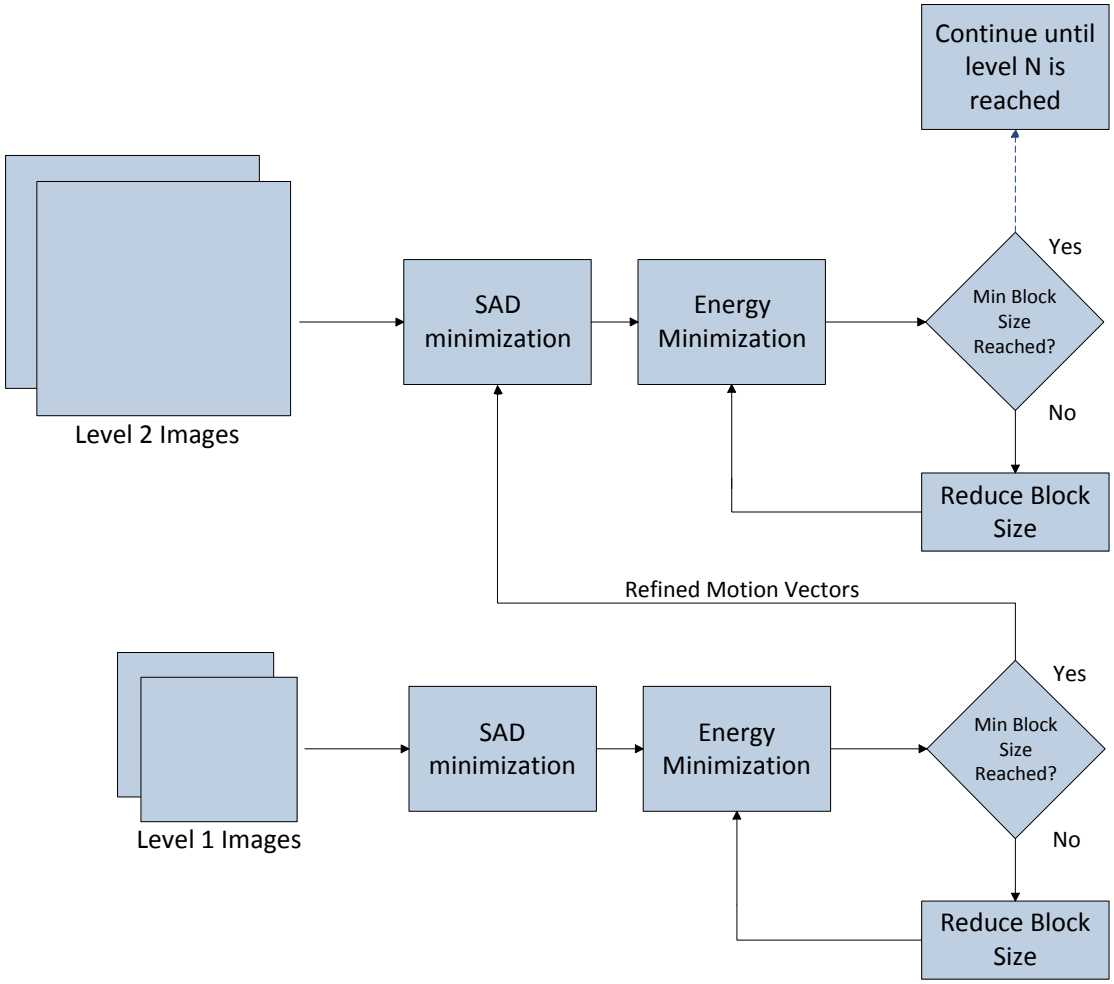


Figure 3.9. Block diagram for preliminary motion estimation algorithm.

As shown in Fig. 3.8 and Fig. 3.9, the motion estimation process begins at the lowest-resolution level of the hierarchy ("Level 1 Images"). Using SAD minimization only, a 128x128 region is searched to determine which 32x32 block is the best match with respect to the SAD only. Once a MV has been determined for each block, the overall energy is minimized with respect to the SAD and spatial MVs. Next, the block size is reduced, and the MVs of the 16x16 blocks take on the MV of the 32x32 block. Energy minimization is then performed for the 16x16 blocks. When the block size becomes 1x1, the refined MVs are then passed to the next level of the hierarchy to initialize the search for the best matching block. At the second level of the hierarchy, the best matching block should be in

the vicinity of the initial estimate from the previous level. Because of this, the search size at the second level only needs to cover a 64x64 region. Similar to the first level, the energy is minimized for each block size until the block size becomes 1x1, and the MVs are then passed up the hierarchy. Since we wish achieve half-pixel accurate MVs for SR, we use a three-level hierarchy.

In the next chapters, we make several modifications to the algorithm of Fig. 3.8 and Fig. 3.9, and we introduce a validity metric that will be used to characterize the "goodness" of the chosen MV. The proposed validity method will be paramount to ensuring that artifacts do not appear in the HR image.

CHAPTER 4

MOTION VECTOR VALIDITY

In this chapter, we first demonstrate that the validity methods in the literature may assign high confidence values to invalid MVs. The failure to assign the correct confidence value to each MV generally stems from the reliance on neighboring MVs and manual thresholds. After demonstrating that these validity metrics are not sufficient, we propose our block-overlap-based validity metric. The block overlap validity metric is directly related to the motion error and does not require manual thresholds or neighboring MVs. To demonstrate that our validity method outperforms other validity metrics, we quantitatively compare the different validity metrics using hybrid de-interlacing. Next, we do a qualitative comparison between HR images generated by SR with different validity metrics, which is the intended application of our motion estimation and validity framework.

To motivate the need for a motion vector (MV) validity method, we begin by showing artifacts that may appear in the HR image if bad MVs are used during the SR reconstruction process. An example using the "Foreman" sequence is shown in Fig. 4.1.

Figure 4.1. Bilinearly interpolated image (left) and artifacts in image generated by SR (right).

In Fig. 4.1, a comparison is shown between a bilinearly interpolated image and the HR image generated by SR. The circles in the right HR image show the areas where artifacts

have been introduced as a result of bad MVs.

4.1 Validity Metrics

In this section, we provide an overview of existing validity metrics that have been applied to block-based motion estimation algorithms and introduce the proposed block-overlap-based validity metric. We divide the existing validity metrics into two categories: smoothness-based validity and gradient/variance-based validity.

4.1.1 Smoothness-Based Validity

4.1.1.1 Smoothness Metric 1

Wang et al. introduced smoothness-based validity in the context of de-interlacing [2]. A confidence value was assigned to each MV based on the block correlation and MV smoothness. Specifically, the validity expression was given as

$$R(\mathbf{v}) \approx \frac{Corr(\mathbf{x}, \mathbf{x}+\mathbf{v}) + Smooth(\mathbf{v})}{\sum_{k \in \mathbf{V}^4} [Corr(\mathbf{x}, \mathbf{x}+\mathbf{v}_k) + Smooth(\mathbf{v}_k)]}, \tag{4.1}$$

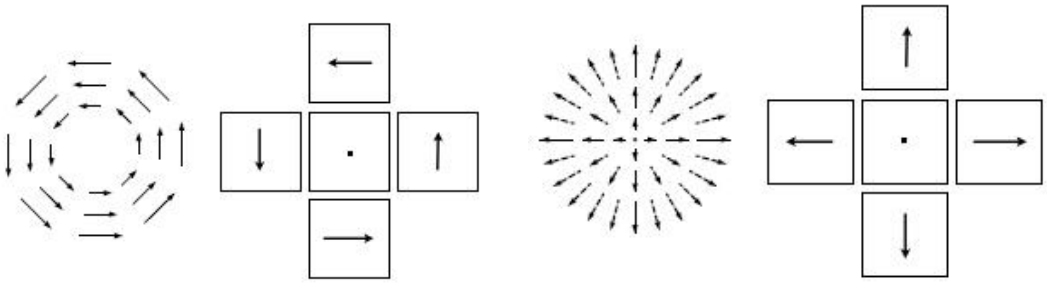
where $\mathbf{v}$ is the current MV, and $\mathbf{x}$ is the position in a block B of pixels. In (4.1), the correlation and smoothness values were calculated for all of the MVs in a four-connected neighborhood, $\mathbf{V}^4$. The $Corr$ and $Smooth$ terms were given as

$$Corr(\mathbf{x}, \mathbf{x}+\mathbf{v}) \approx \sum_{\mathbf{x} \in B} [I_1(\mathbf{x}) - I_0(\mathbf{x}+\mathbf{v})]^2 \tag{4.2}$$

$$Smooth(\mathbf{v}) \approx \min \left\{ \|\mathbf{v} - \mathbf{v}_k\|^2 \,\middle|\, 0 < k \leq 4 \right\}, \tag{4.3}$$

where I_1 and I_0 are the current and previous images, respectively.

The $Smooth$ term in (4.3) assumes that a true MV must be similar to at least one of its neighboring MVs in order to be valid. However, as shown in Fig. 4.2, this assumption fails in the event of a rotation or scaling, where the neighboring MVs are significantly different from the center MV.



(a) MVs for rotation. (b) MVs for scaling.

Figure 4.2. Center and neighboring MVs for rotation and scaling.

4.1.1.2 Smoothness Metric 2

A similar smoothness-based validity metric was introduced by Liu and Shen in the context of super-resolution [76]. Liu and Shen made the assumption that a MV is valid if it results in a small normalized SAD and has consistent neighboring MVs. However, in contrast to the work of Wang et al., thresholds were applied to both the normalized SAD and smoothness in order to classify a MV as valid or invalid. The decision rule was given as

$$
\begin{aligned}
&If \quad (Corr(\mathbf{x}, \mathbf{x}+\mathbf{v}) < T_E) \mapsto R(\mathbf{v}) = 1 \\
&Else \\
&\qquad If\ (Smooth(\mathbf{v}) < T_C) \mapsto R(\mathbf{v}) = 1 \\
&\qquad Else \mapsto R(\mathbf{v}) = 0 \\
&\qquad End \\
&End,
\end{aligned}
\tag{4.4}
$$

where T_E and T_C were determined experimentally. More details can be found in [76]. The $Corr$ and $Smooth$ terms were given as follows:

$$
Corr(\mathbf{x}, \mathbf{x}+\mathbf{v}) = \frac{\sum_{\mathbf{x}\in B} \left| I_1(\mathbf{x}) - I_0(\mathbf{x}+\mathbf{v}) \right|}{\epsilon + \sum_{\mathbf{x}\in B} I_1(\mathbf{x})} \tag{4.5}
$$

$$
Smooth(\mathbf{v}) = \sum_{k\in\mathbf{V}^8} \mathbf{a}(\hat{\mathbf{v}} \cdot \hat{\mathbf{v}}_k), \tag{4.6}
$$

where $\mathbf{a}(\hat{\mathbf{v}} \cdot \hat{\mathbf{v}}_k)$ is an indicator function for the dot product of unit vectors $\hat{\mathbf{v}}$ and $\hat{\mathbf{v}}_k$, and ϵ is a small scalar to prevent division by zero. The neighborhood size was increased from $\mathbf{V}^4$ to $\mathbf{V}^8$, which includes horizontal, vertical, and diagonal neighbors. The indicator function was given as

$$\mathbf{a}(\hat{\mathbf{v}} \cdot \hat{\mathbf{v}}_k) = \left\{ \begin{array}{ll} 1 & \hat{\mathbf{v}} \cdot \hat{\mathbf{v}}_k > T_S \\ 0 & \text{otherwise} \end{array} \right\}, \tag{4.7}$$

and the threshold T_S was determined experimentally.

Aside from the difficulty in setting thresholds, this metric also assumes that a MV is valid if the *Corr* term is below a given threshold, which fails for uniform regions. In addition, the *Smooth* term in (4.6) is not a useful measure of validity for motion such as rotation and scaling, which was shown in Fig. 4.2.

4.1.2 Gradient/Variance-Based Validity

The two metrics in this section rely on block texture in addition to a correlation metric (SAD or square displaced frame difference (DFD)). While the SAD or square DFD rely on the intensity differences between pixels in a block, the texture of a block provides a characterization of more intuitive qualities such as coarseness, contrast, directionality, line-likeness, and roughness [77]. In the context of validity, properly discerned texture provides an additional discriminator for determining if image blocks are a good match.

4.1.2.1 Gradient Metric

François et al. calculated a confidence value for a given direction which depends on the spatial gradient in that direction and on the DFD [78]. The spatial gradient, which is commonly approximated using a Sobel operator, is a computationally inexpensive measure of texture. In the work of [78], the validity was given separately for the $x-$ and $y-$directions. The validity for the x-direction was given as

$$R_x(\mathbf{v}) = \frac{1}{1 + \dfrac{1 + Corr(\mathbf{x}, \mathbf{x} + \mathbf{v})}{2 Text_x(\mathbf{x})}}. \tag{4.8}$$

In (4.8), the *Corr* term is the same as that given in (4.2), and the *Text* term was given as follows:

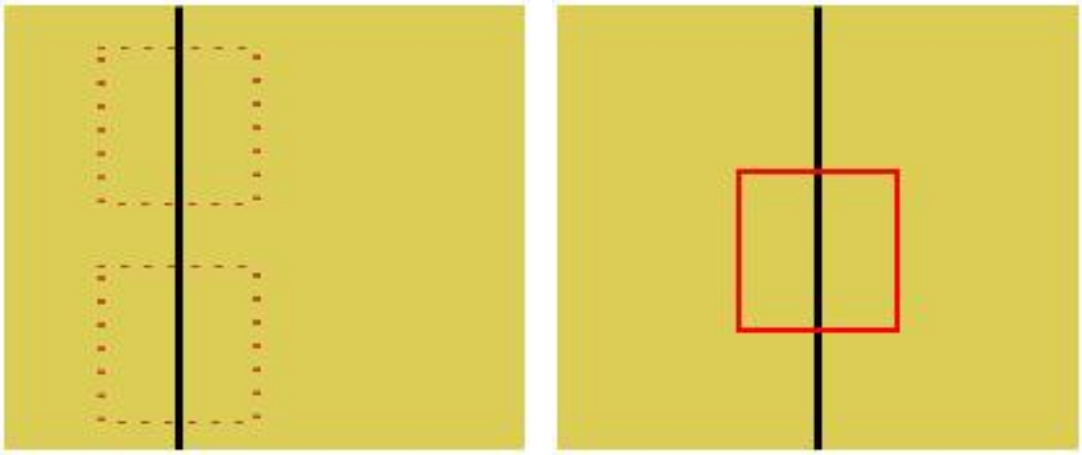
$$Text_x(\mathbf{x}) = \sum_{\mathbf{x} \in B} G_x^2(\mathbf{x}), \tag{4.9}$$

where $G_x(\mathbf{x})$ is the gradient in the x-direction for each position $\mathbf{x}$ within block B. Similarly, $R_y(\mathbf{v})$ can be found by replacing x with y in (4.8) and (4.9). However, since a single confidence value for each MV is desired, we make a slight modification to the metric of François et al. by bounding the validity such that

$$R(\mathbf{v}) = \min(R_x(\mathbf{v}), R_y(\mathbf{v})). \tag{4.10}$$

Therefore, the validity of any given MV will be bounded by the least valid component of the MV.

The expression given in (4.8) implies that a large confidence value should be assigned to a MV that has a large gradient and a small DFD. However, this is not true around edges, where the DFD may be small and the gradient may be large. As an example, consider the two images with a vertical edge as shown in Fig. 4.3.



(a) Matches in adjacent image. (b) Desired block in current image.

Figure 4.3. Multiple matches with a large gradient and small DFD.

In Fig. 4.3, the DFD will be small for either of the block matches in Fig. 4.3(a), and the block in Fig. 4.3(b) contains an edge (and hence large gradient). Therefore, this metric

may incorrectly assign high confidence values to invalid MVs.

4.1.2.2 Variance Metric

Patti et al. introduced a variance-based validity metric in the context of de-interlacing [1]. An adaptive threshold-based metric was used that assumes regions of low local variance should have a low SAD threshold value, whereas regions of high local variance should have a high SAD threshold value. The entire process is shown in Fig. 4.4.

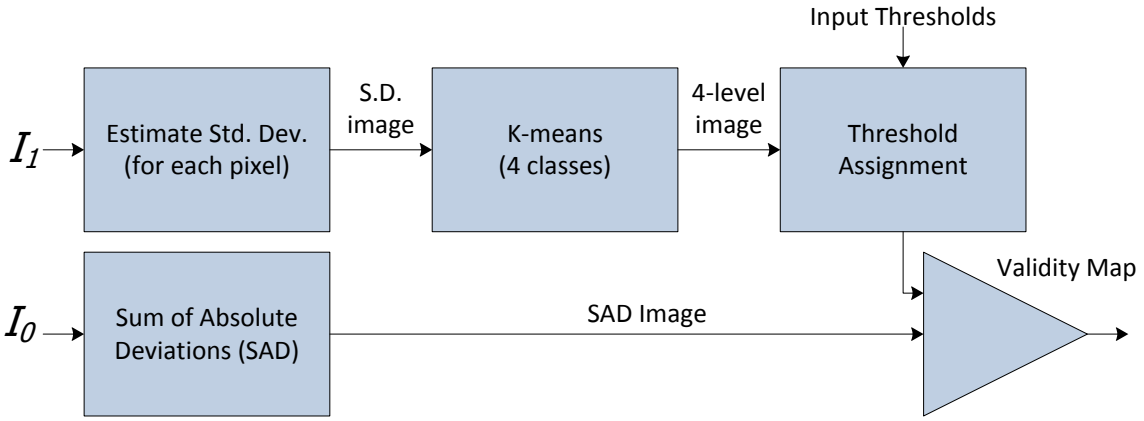


Figure 4.4. Generation of validity map for the metric of [1].

Starting from the image pair $\{I_1, I_0\}$, the standard deviation is estimated for each pixel in image I_1, which produces the standard deviation (SD) image. To remove high SD values, the SD image is quantized into four classes using the k-means algorithm [79]. Next, a set of predetermined image thresholds are assigned to each of the four classes. The validity map is determined by comparing the SAD value to the standard deviation thresholded value. If the SAD value is below its corresponding deviation threshold, the MV is labeled as valid. Otherwise, the MV is labeled invalid.

The main difficulty with the process shown in Fig. 4.4 is the determination of the 'Input Thresholds'. These thresholds were determined experimentally in [1], and were shown to be sensitive to image content. In the results of [1], it was also demonstrated that this metric fails around stationary edges in the image.

4.1.3 Proposed Validity Metric

A block-overlap-based validity metric is proposed to overcome the weaknesses of the metrics in Chapter 4.1.1 and Chapter 4.1.2, where it was shown that the validity metrics were sensitive to neighboring MVs, image features, and thresholds.

To motivate our block overlap approach, we begin by considering a pair of images whose motion we wish to estimate. Let the current image I_1 be divided into a grid of square blocks of size B_S, and let the grid of the adjacent image I_0 be determined by the MC blocks, as shown in Fig. 4.5. The position of a block in I_1 is denoted as $\mathbf{x}$ and the position of the MC block in I_0 as $\mathbf{y} = \mathbf{x} + \mathbf{v}$, where $\mathbf{v}$ is the MV. To simplify the analysis, we assume a square grid; however, the analysis also holds for non-square grids.

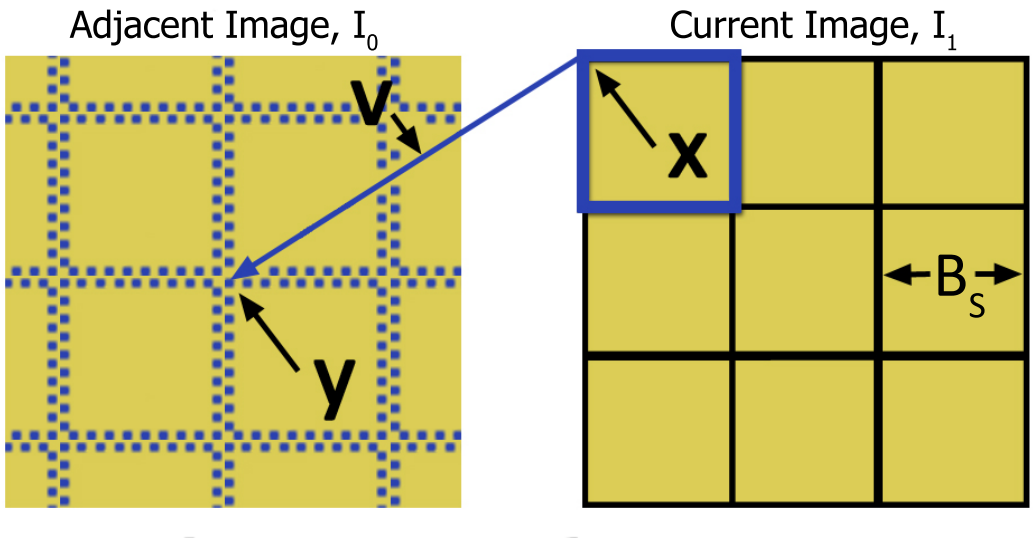


Figure 4.5. Blocks in current image I_1 and MC blocks in adjacent image I_0.

In the ideal case, there exists an injective function $f : I_1 \mapsto I_0$ such that each block in I_1 is mapped to a unique block in I_0. However, for real images with various motion types, the process of mapping blocks from I_1 to I_0 is non-surjective, i.e., blocks in I_1 may be mapped to same block position in I_0, and the whole of I_0 is not necessarily filled. An example of the block mapping for real images is shown in Fig. 4.6.

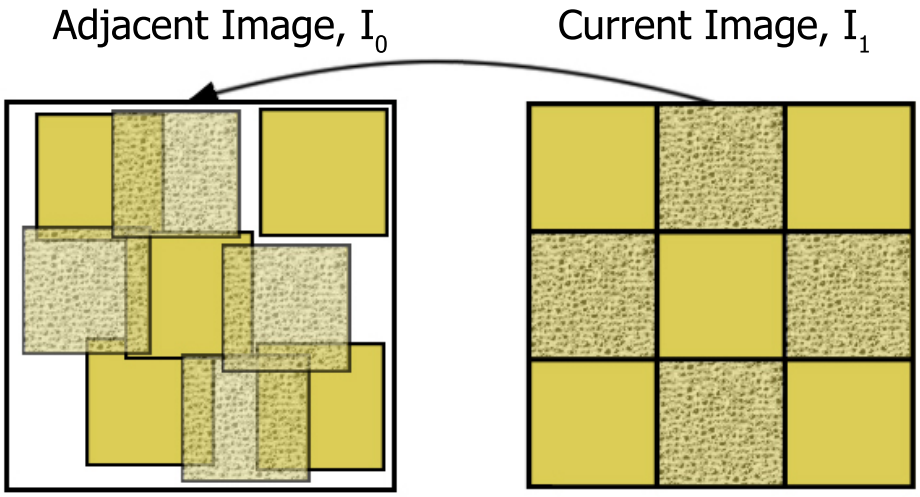


Figure 4.6. Block mappings for real images.

As shown in Fig. 4.6, the mapped MC blocks for real images may overlap each other to different degrees. We wish to characterize the degree of overlap as the uncertainty in the motion estimation decision, i.e., the validity of a given MV will depend on the amount to which its MC block overlaps with other MC blocks.

We now derive the proposed validity metric. An MC block at position $\mathbf{y}$ will cover an area of B_S^2 in I_0 if no overlap occurs. However, when blocks in I_0 overlap, a volume (perhaps nonuniform) is generated, which we denote as $\mathbf{L_x}(\mathbf{y})$, the overlap at position $\mathbf{y}$ mapped from position $\mathbf{x}$. The degree of overlap for the block at position $\mathbf{y}$ is given by the ratio $\dfrac{B_S^2}{\mathbf{L_x}(\mathbf{y})}$, where a large ratio indicates less/no overlap and therefore a higher confidence in MV $\mathbf{v}$.

However, in some cases the block at position $\mathbf{y}$ does not generate a large degree of overlap in I_0 even though the block has a large SAD value. Therefore, it is necessary to weight the degree of overlap by the SAD value. The validity metric for MV $\mathbf{v}$ is given as

$$R(\mathbf{v}) = \frac{B_S^2}{\left(1 + \dfrac{\mathrm{SAD}(\mathbf{x},\mathbf{y})}{\mu}\right)\mathbf{L_x}(\mathbf{y})}, \tag{4.11}$$

where SAD($\mathbf{x}$,$\mathbf{y}$) is the SAD between blocks $\mathbf{x}$ and $\mathbf{y}$, and μ is the mean of the SAD value

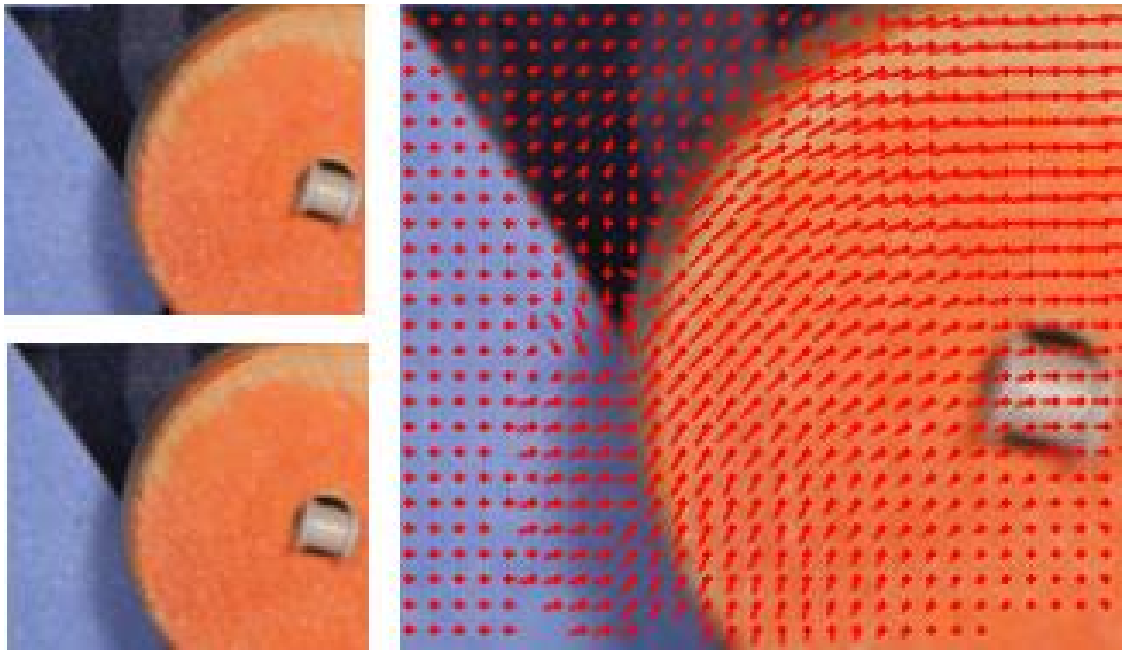
over all MVs. The mean μ is included to make the validity metric more robust to brightness variations between images. Equation (4.11) states that a MV (and hence MC block) which does not produce any overlap in I_0 and has a small SAD value will result in a large confidence value. The algorithm used to determine $\mathbf{L_x}(\mathbf{y})$ is given in Fig. 4.7, where y_i and y_j represent the vertical and horizontal position of the block at position $\mathbf{y}$, respectively.

Algorithm 2 : Calculate volume for each $\mathbf{L_x}(\mathbf{y})$

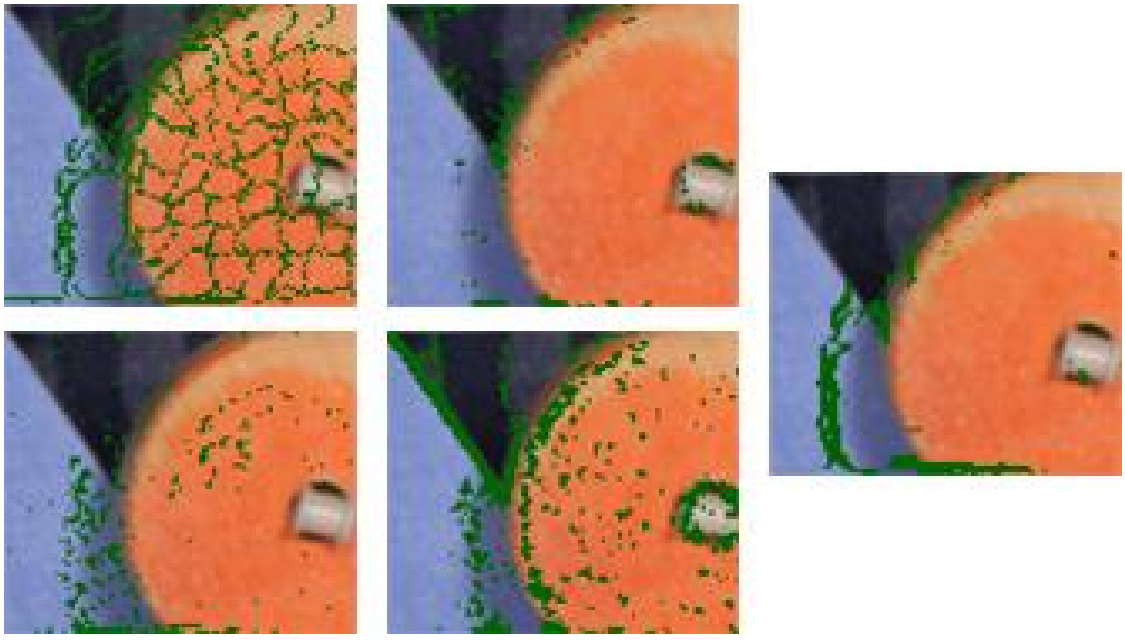
For all blocks in I_1, set $\mathbf{y} = \mathbf{x} + \mathbf{v}$
for $k = y_i,\ k < y_i + B_S,\ k$++ **do**
  for $l = y_j,\ l < y_j + B_S,\ l$++ **do**
    $\mathbf{L_x}\begin{bmatrix} k \\ l \end{bmatrix}$ += 1
  end for
end for

Figure 4.7. Algorithm for calculating block overlap volume.

We demonstrate the strength of the proposed validity metric using an image pair from [3] (top left/bottom left) and the corresponding MV field (right), as shown in Fig. 4.8(a).



(a) Current/Adjacent Images and corresponding MVs.



(b) Pixels with $R(\mathbf{v}) < 0.5$ for different validity metrics.

Figure 4.8. Image Pair, MVs, and invalid pixels for "Army" sequence.

The images in Fig. 4.8(b) show the MVs in I_1 with low confidence values for the different validity metrics. The smoothness-based metric of Wang (top left image of Fig. 4.8(b))

penalizes the minimum MV deviation, which can be seen to mark valid MVs on the rotating disk as invalid. The smoothness-based metric of Liu (top right of Fig. 4.8(b)) shows an improvement for the MVs on the rotating disk, but it does not detect the occlusion for the blue surface to the left of the disk. The gradient-based (bottom left of Fig. 4.8(b)) and variance-based (bottom right) metrics also do a poor job of classifying MVs as valid or invalid. The proposed block overlap metric (far right of Fig. 4.8(b)), however, correctly detects occlusion and invalid MVs surrounding the disk.

4.2 Validity Metric Comparison using Hybrid De-Interlacing

To quantitatively compare our validity metric with the metrics introduced in the previous sections, we implemented the hybrid de-interlacing algorithm of [2]. The algorithm discussed in [2] chooses between two types of de-interlacers based on the validity of MVs. The first de-interlacer is based on the generalized sampling theorem (GST) [80][81], which is theoretically the optimal method for generating a de-interlaced image. However, this method is very sensitive to MV errors and often produces artifacts in the de-interlaced image. The second de-interlacer is based on the vertical-temporal filter with motion compensation (MC V-T filter) [81], which is very robust to MV errors. Although this method is less prone to generating artifacts, it produces lower-resolution de-interlaced images than those of the GST de-interlacer.

The choice between the two de-interlacing methods is made based on the MV confidence value, which is illustrated in Fig. 4.9. The confidence value is used to weight pixels of both image $G(\mathbf{x})$ and $F(\mathbf{x})$ in order to produce the final progressive image, $H(\mathbf{x})$.

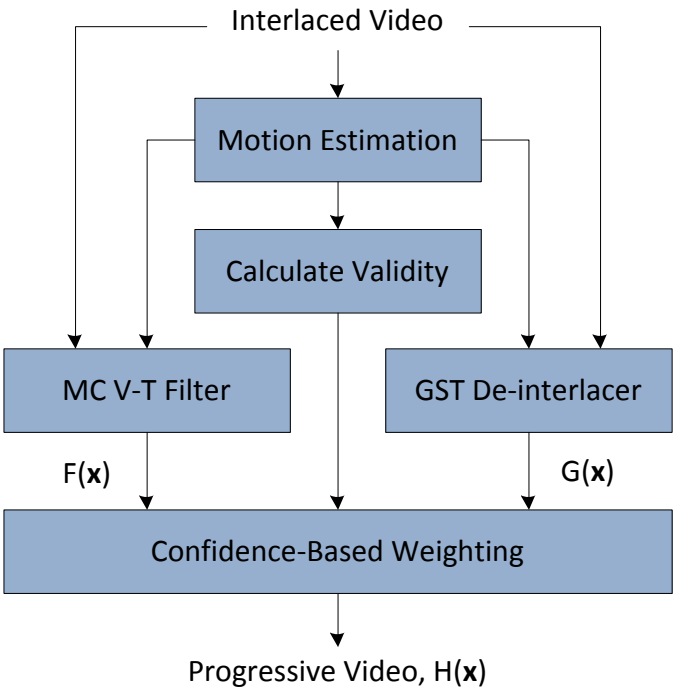


Figure 4.9. Hybrid de-interlacing algorithm of [2].

Using the block diagram shown in Fig. 4.9, the proposed validity metric was quantitatively compared with the four other validity metrics. The MVs were estimated to quarter-pixel accuracy using a hierarchical block matching algorithm, and a 3x3 block size was used for all validity metrics.

Interlaced test sequences were created from the following five progressive video sequences: *Flower Garden, Football, Foreman, Mobile Calendar,* and *Suzie*. To avoid aliasing, local pixel averaging was used to generate the interlaced images from the original progressive images. The PSNR of the de-interlaced image was calculated with respect to the original progressive image for all frames of the five sequences. The PSNR (H(**x**) in Fig. 4.9) of each sequence for the different validity metrics is shown in Fig. 4.10.

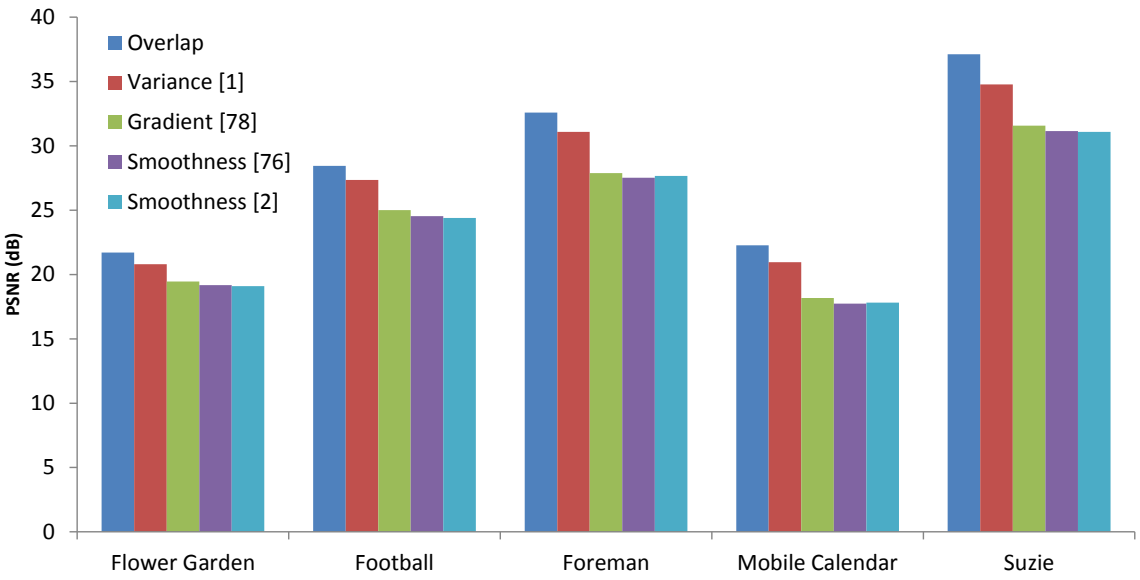


Figure 4.10. Average PSNRs of hybrid results for different validity metrics. To visualize the results more easily: for each sequence, the top to bottom ordering in the legend matches the left to right ordering of the bars.

It can be seen that the proposed validity metric outperforms the four other validity metrics for all of the video sequences. The smoothness-based metrics performed similarly, and the gradient-based metric only provided a slight improvement over the smoothness-based metrics. The variance-based metric was a significant improvement; however, when averaged over all of the video sequences, the proposed metric provided an additional 1.4 dB of improvement over the variance-based metric.

4.3 Validity Metric Comparison using Super-Resolution

In this section, we qualitatively compare two validity metrics from the literature and the proposed validity method in the context of super-resolution. To do so, we first give the POCS-based SR algorithm that will be used to merge the LR images into one HR image. The HR image generated by SR will be shown with and without a validity filter in order to demonstrate the need for artifact reduction without sacrificing the quality of the image.

The steps necessary to generate an HR image using the POCS method are shown in

Fig. 4.11 and can be summarized as follows:

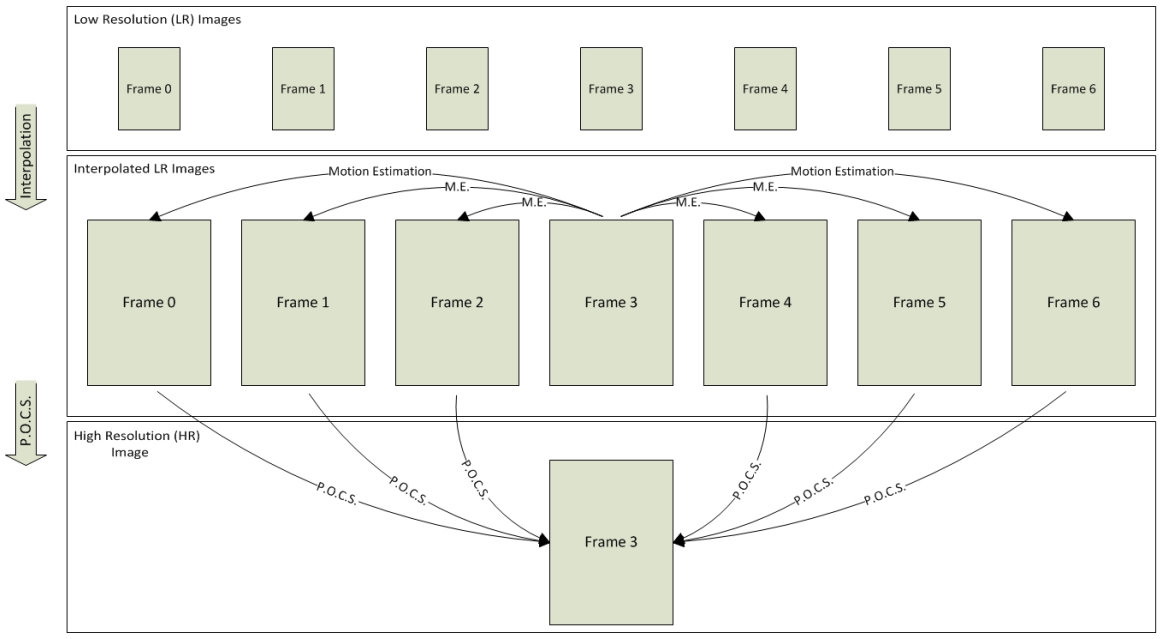


Figure 4.11. SR process diagram to generate HR image.

1. Buffer 7 to 15 frames of the video sequence for a 2x magnification in resolution. The number of frames needed for a 2x magnification depends on the video sequence, but 7 to 15 frames is generally sufficient. For greater magnification factors, more frames are needed.

2. Choose the center frame of the 7 or 15 frame sequence as the "anchor" frame. All MVs will be computed with respect to this frame.

3. Interpolate the anchor frame and each frame in the sequence to perform motion estimation. We use bilinear interpolation for this step; however, we note that there more computationally efficient methods than interpolation [61][62][63][64].

4. Use the estimated MVs for each frame to perform POCS reconstruction on the interpolated anchor frame. The interpolated anchor frame is taken as the initial solution in the POCS reconstruction. We assume a spatially-invariant PSF and only include the data consistency and amplitude constraints (see Chapter 2.5.4).

4.3.1 Results without MV Validity Metrics

To demonstrate the results of the SR algorithm without a MV validity metric, we use the well-known "Foreman" sequence [82]. The seven frames for the "Foreman" sequence are shown in Fig. 4.12.

(a) Frame 0. (b) Frame 1. (c) Frame 2. (d) Frame 3.

(e) Frame 4. (f) Frame 5. (g) Frame 6.

Figure 4.12. Seven frames of "Foreman" sequence.

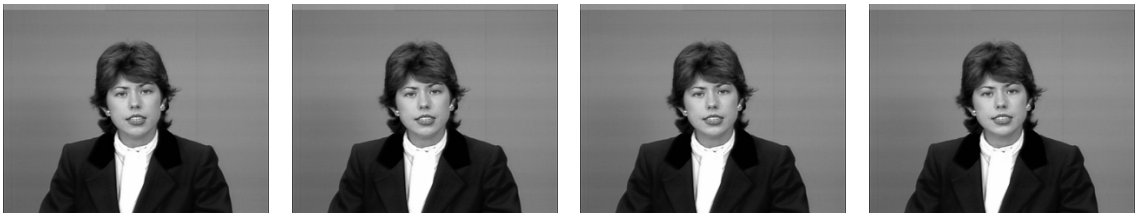
The interpolated anchor frame and the HR image generated by SR reconstruction are shown in Fig. 4.13(a) and Fig. 4.13(b), respectively.

(a) Interpolated image. (b) Super-Resolution HR image.

Figure 4.13. Comparison between interpolated image and SR-generated image.

In Fig. 4.13(b), more details can be seen in the face of the man. However, there are also artifacts introduced around the contour of the face and hard hat. These artifacts are a result of MV errors, specifically from the occlusion of foreground and background. We return to this issue in the next section, where the proposed MV validity metric will be applied when generating the HR image in Fig. 4.13(b).

We further demonstrate the SR algorithm without MV validity on the "Claire" sequence. The seven frames for the sequence are shown in Fig. 4.14.



(a) Frame 0. (b) Frame 1. (c) Frame 2. (d) Frame 3.

(e) Frame 4. (f) Frame 5. (g) Frame 6.

Figure 4.14. Seven frames of "Claire" sequence.

The interpolated anchor frame and the HR image generated by SR reconstruction are shown in Fig. 4.15(a) and Fig. 4.15(b), respectively.

(a) Interpolated image. (b) Super-Resolution HR image.

Figure 4.15. Comparison between interpolated image and SR-generated image.

In the "Claire" sequence, unlike the "Foreman" sequence, there are only minimal errors introduced as a result of the small displacements between frames.

4.3.2 Results with MV Validity Metrics

In this section, we demonstrate two validity methods from the literature and show that these methods are insufficient for eliminating artifacts in the HR image. We then show that the proposed validity metric eliminates artifacts without reducing the quality of the HR image.

The first validity method uses a threshold value on the SAD metric [46][83][84][85][86][87]. This method is demonstrated on the same frames of the "Foreman" sequence as in the previous section. Note that the HR image shown in the previous section has not undergone any filtering to remove errors. The following rule is used to threshold the SAD value:

$$
\begin{aligned}
&if\ (\mathrm{SAD}(B_{m,n}) > Th) \\
&\qquad \text{Skip } \forall \mathbf{x} \in B_{m,n} \text{ in POCS} \\
&else \\
&\qquad \text{Include } \forall \mathbf{x} \in B_{m,n} \text{ in POCS,}
\end{aligned}
\tag{4.12}
$$

where $\mathrm{SAD}(B_{m,n})$ is the SAD value of the block positioned at pixel location $\mathbf{x}[m,n]$ and Th is a fixed threshold value. The rule given by (4.12) says that only those pixels that come

from a block with a SAD value below the threshold are used in the POCS reconstruction. A block size of 2x2 is used for the rule of (4.12).

The result of using (4.12) on the "Foreman" sequence is shown in Fig. 4.16. The unfiltered HR image generated by SR is shown in Fig. 4.16(a) and the SAD-filtered HR image generated by SR is shown in Fig. 4.16(b). The threshold value which removed the largest number of artifacts without significant loss of detail in the HR image was chosen.

(a) HR image after SR. (b) HR image after SR with SAD filtering.

Figure 4.16. Result of using SAD filter with SR.

As shown in Fig. 4.16, using the SAD threshold clears up some of the noise around the edges of the man's silhouette. However, a large amount of noise is still present in the image.

Next, we demonstrate the method of using the SAD metric along with MV smoothness [88][89]. We write the rule for this method as follows:

$$
\begin{aligned}
&if\ ((\text{SAD}(B_{m,n}) + \lambda\ \text{Smoothness}(V_{m,n})) > Th) \\
&\quad \text{Skip } \forall \mathbf{x} \in B_{m,n} \text{ in POCS} \\
&else \\
&\quad \text{Include } \forall \mathbf{x} \in B_{m,n} \text{ in POCS.}
\end{aligned}
\tag{4.13}
$$

The result of using (4.13) on the "Foreman" sequence is shown in Fig. 4.17. The unfiltered HR image generated by SR is shown in Fig. 4.17(a) and the SAD- and Smoothness-filtered HR image generated by SR is shown in Fig. 4.17(b).

(a) HR image after SR.

(b) HR image after SR with SAD and Smoothness filter.

Figure 4.17. Result of using SAD and smoothness filter with SR.

The threshold value which removed the largest number of artifacts without significant loss of detail in the HR image was chosen. As seen in Fig. 4.17(b), the SAD and Smoothness filter is a large improvement over the SAD filter from Fig. 4.16(b). Most of the artifacts around the man's silhouette have been removed, but some still remain around his right ear and hard hat. These artifacts can be removed by setting the threshold to a lower value, however, only at the expense of loss of detail in the HR image.

Therefore, we turn to the proposed validity method to further reduce the artifacts and maintain HR image quality. The proposed validity method is re-written as

$$R(\mathbf{v}) = \frac{B_S^2}{\left(1 + \dfrac{\mathrm{SAD}(\mathbf{x}, \mathbf{y})}{\mu}\right)\mathbf{L}_{\mathbf{x}}(\mathbf{y})}, \tag{4.14}$$

where B_S is the block size, $\mathbf{L}_{\mathbf{x}}(\mathbf{y})$ is the block overlap volume, SAD($\mathbf{x}$,$\mathbf{y}$) is the SAD between blocks $\mathbf{x}$ and $\mathbf{y}$, and μ is the mean of the SAD value over all MVs. The rule for

determining which pixels to include in the SR image reconstruction is given as follows:

$$
\begin{aligned}
&if\ (R(\mathbf{v}) < Th) \\
&\qquad \text{Skip } \forall \mathbf{x} \in B_{m,n} \text{ in POCS} \\
&else \\
&\qquad \text{Include } \forall \mathbf{x} \in B_{m,n} \text{ in POCS,}
\end{aligned}
\tag{4.15}
$$

where a 2x2 block size is used in determining $R(\mathbf{v})$, and all pixels in the 2x2 blocks are considered valid or invalid based on this measure.

The result of using (4.15) on the "Foreman" sequence is shown in Fig. 4.18. The unfiltered HR image generated by SR is shown in Fig. 4.18(a) and the proposed validity-filtered HR image generated by SR is shown in Fig. 4.18(b).

(a) HR image after SR (unfiltered). (b) HR image after SR with proposed validity filter.

Figure 4.18. Result of using proposed validity filter with SR.

The threshold value which removed the largest number of artifacts without significant loss of detail in the HR image was chosen, and experimental results shown that a threshold value of 0.25 provides the best trade-off between reducing artifacts and preserving detail.

With the proposed validity filter, it can be seen the artifacts are reduced in Fig. 4.18. Specifically, artifacts around the edges of the hard hat and around the contour of the face in the image of Fig. 4.18(a) have been removed in Fig. 4.18(b).

In the next chapter, we use the proposed validity method as a regularizer to improve the quality of the motion field.

CHAPTER 5

BLOCK-OVERLAP-BASED REGULARIZER

Although regularizing the motion estimation problem using the smoothness constraints of Chapter 3.3 reduces the number of possible solutions, it does not force a unique solution. In fact, unless the regularization function is strictly convex, multiple solutions may exist. In previous approaches [65][66][70], the chosen solution depends on the block matching search order and on the order in which the smoothness constraints are applied. To help overcome this, we introduce the block-overlap-based validity metric from the previous chapter as a regularizer to improve the quality of the motion field.

5.1 Problem Formulation

The motion estimation problem was formulated as an energy minimization problem in Chapter 3.1, which re-write below as

$$E = \min_i \{\mathcal{D}(I_0, I_1, v_i) + \lambda \mathcal{R}(v_i)\}, \tag{5.1}$$

where $\mathcal{D}(I_0, I_1, v_i)$ is a data term that measures the similarity of blocks in images I_0 and I_1 for a given MV v_i, $\mathcal{R}(v_i)$ is a regularization term which penalizes deviations in the smoothness of the motion field, and the Lagrange multiplier λ is used to weight the regularization term over the data term. The goal of the motion estimation problem is to choose a MV v_i such that the energy in (5.1) is minimized.

5.2 Shortcomings of Previous Work

As previously discussed, there will not necessarily be a unique minimum for a given MV since neither of the terms in (5.1) is strictly convex. Therefore, it is necessary to choose between MVs that produce the same overall energy. Without an explicit way to discriminate between MVs that produce the same overall energy, the chosen MV will depend on the

order in which the spatial candidates are tested in (5.1). Fortunately, as we will show in the next section, a block overlap regularizer helps to choose the optimal MV in such cases.

5.3 Block Overlap Minimization Framework

Block overlap introduces an additional error metric for determining which MV minimizes the overall energy. We modify the energy expression of (5.1) as follows:

$$E = \min_i \{\mathcal{D}(I_0, I_1, v_i) + \lambda \mathcal{R}(v_i) + \mathcal{O}(I_0, I_1, v_i)\}, \tag{5.2}$$

where $\mathcal{O}(I_0, I_1, v_i)$ represents the overlap regularizer for the MC block in I_0 given by v_i. Similar to the overlap-based validity metric in Chapter 4.1.3, the overlap regularizer is given as

$$\mathcal{O}(I_0, I_1, v_i) = (1 + \mathcal{D}(I_0, I_1, v_i)) \frac{\mathbf{L_x(y)}}{B_S^2}, \tag{5.3}$$

where $\mathbf{L_x(y)}$ is the overlap volume given in Fig. 4.7, and B_S is the block size. In (5.3), the overlap volume $\frac{\mathbf{L_x(y)}}{B_S^2}$ is multiplied by the data term $\mathcal{D}(I_0, I_1, v_i)$, which allows the overlap volume to contribute more to the energy for large SAD values and less for small SAD values. We add the constant '1' to the data term so that the overlap volume still contributes to the energy when the SAD value is zero.

We re-write the SAD and smoothness constraints from Chapter 3.1 in (5.4) and (5.5), respectively.

$$\mathcal{D}(I_0, I_1, v_i) = \sum_{\mathbf{x} \in B} |I_0(\mathbf{x}) - I_1(\mathbf{x} + v_i)|, \tag{5.4}$$

$$\mathcal{R}(v_i) = \sum_{j \in C^s} \left\| v_i - v_j \right\|_1, \tag{5.5}$$

Using the SAD (5.4), the smoothness constraints (5.5), and the overlap regularizer (5.3), the energy expression of (5.2) can be rewritten as

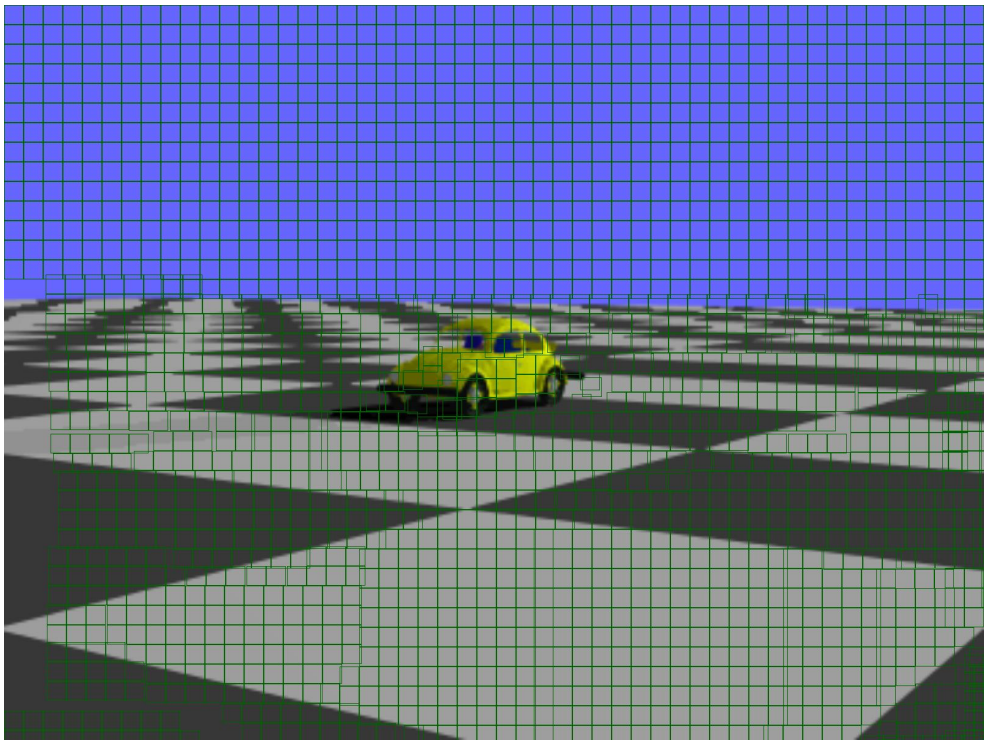
$$E = \min_i \left\{ \left(1 + \frac{\mathbf{L_x}(\mathbf{x} + v_i)}{B_S^2}\right) \left(1 + \sum_{\mathbf{x} \in B} |I_0(\mathbf{x}) - I_1(\mathbf{x} + v_i)|\right) + \lambda \sum_{j \in C^s} \left\| v_i - v_j \right\|_1 \right\} \tag{5.6}$$

In the next two sections, we demonstrate some of the advantages of the new energy minimization framework of (5.2) compared to the framework of (5.1).

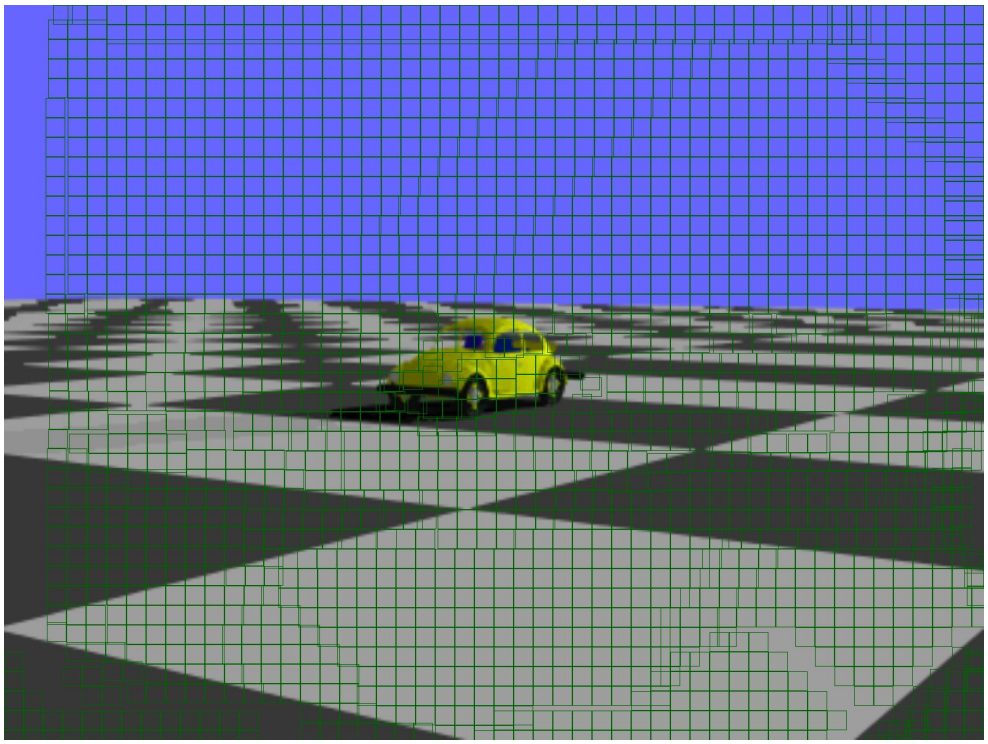
5.4 Uniform Block Distribution in Textureless Regions

One of the advantages of the overlap regularizer is that it provides a more uniform distribution of blocks in regions with little texture (smooth regions). In general, smooth regions will have multiple minima, and without the block overlap regularizer, the value of λ should be large to force the MVs to converge. However, without knowing the segmentation of the objects in the image, a large value of λ will also oversmooth the MVs in other regions.

If any two MVs have the same combined SAD and smoothness, the overlap regularizer in (5.6) will choose the MV (and hence MC block) which results in the least amount of overlap. As shown in Fig. 5.1, the new energy framework of (5.6) provides a more uniform distribution of blocks for the smooth regions than the energy framework of (5.1).



(a) MC blocks using new energy of (5.6).



(b) MC blocks using energy of (5.1).

Figure 5.1. Visual comparison of MC blocks for overlap and non-overlap versions of energy equation.

5.5 Reduced Sensitivity to Block Size

Another advantage of the overlap regularizer is reduced sensitivity to block size. The reduced sensitivity is closely related to the distribution of blocks; the overlap regularizer attempts to keep the distribution of blocks uniform regardless of block size. To demonstrate the reduced sensitivity, we chose the "Grove 2" image sequence from the Middlebury database [3] and varied the block size for the energy minimization frameworks of (5.6) and (5.1). The change in MV error for different block sizes is shown in Fig. 5.2.

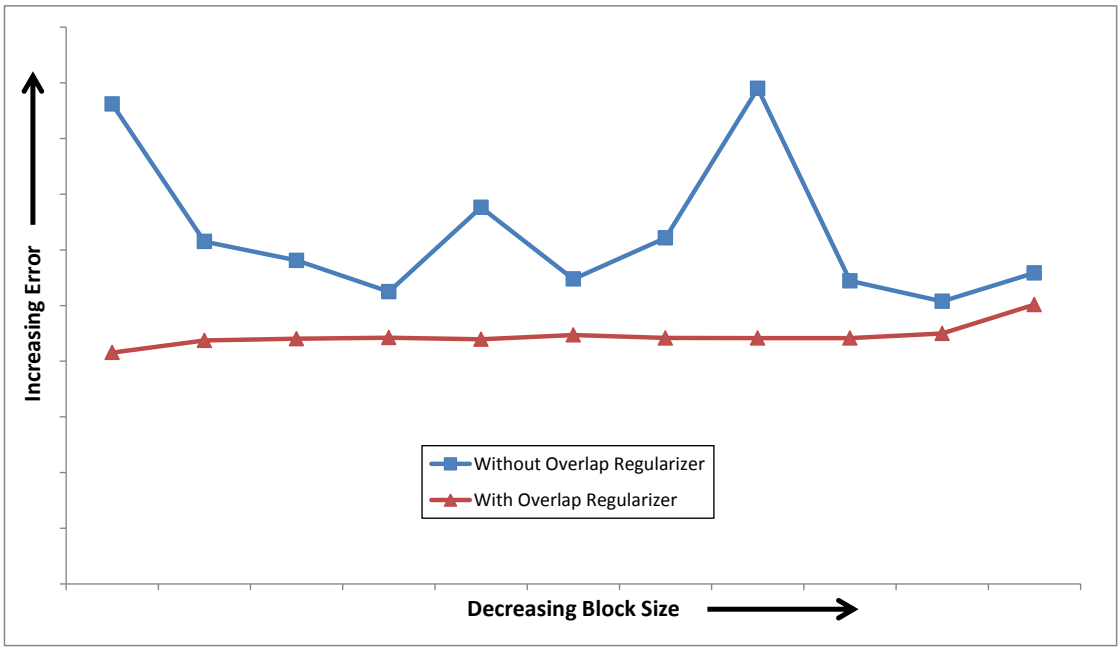


Figure 5.2. MV error for different block sizes.

5.6 Progression of Algorithm

The progression of the algorithm used to minimize the energy of (5.6) is given in Algorithm 1 of Fig. 5.3.

Algorithm 1 : Preliminary Motion Estimation Algorithm

1: Form image hierarchy and begin at lowest-resolution level.
2: For all blocks in image I_1, find the corresponding blocks in image I_0 with the lowest SAD error.
3: Using the MVs from Line 2, find the MV with the minimum energy.
4: Iterate Line 3 until MVs converge, reduce the block size, and repeat until block size is 1x1.
5: Pass converged MVs to next level of hierarchy and return to Line 2. Repeat until highest-resolution level of hierarchy is reached.

Figure 5.3. Algorithm 1 - Preliminary Motion Estimation Algorithm

The block diagram for the steps in Fig. 5.3 is given in Fig. 5.4.

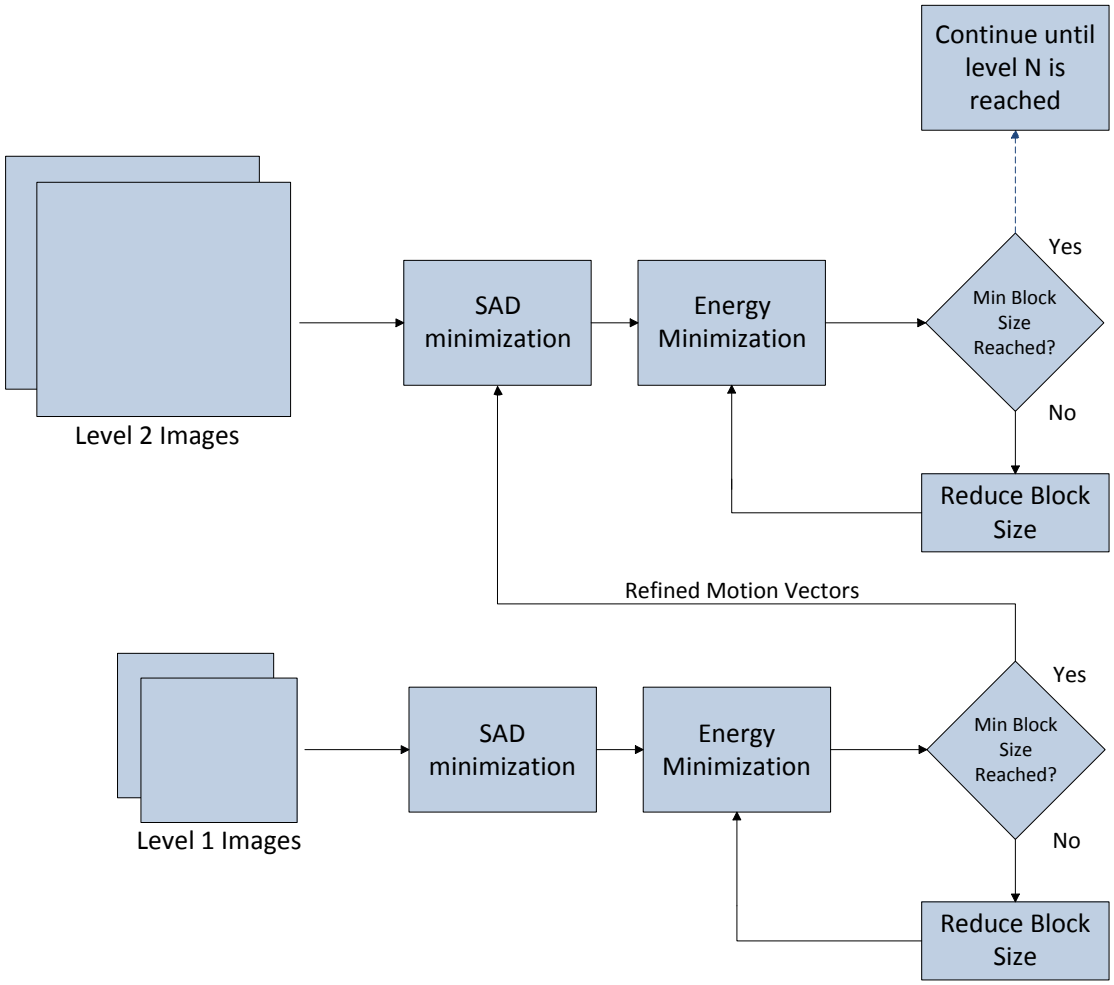


Figure 5.4. Block diagram for preliminary motion estimation algorithm.

5.7 Results

In the results that follow, we used a four-level hierarchy and the algorithm given in Fig. 5.3 to obtain quarter-pixel MVs. For images with a VGA resolution, the run time of the proposed method is approximately two seconds using unoptimized code.

We demonstrate the effectiveness of the new energy minimization framework using the eight ground truth test sequences from Middlebury University [3]. In Table 7, we show comparisons of endpoint error for the MVs of (5.6) and (5.1).

Table 7. Improvement of new energy minimization (5.6) **over** (5.1)**.**

Sequence	**Endpoint Error for** (5.6)	**Endpoint Error for** (5.1)	**Improvement in dB**
Dimetrodon	0.215	0.215	0.00 dB
Grove 2	0.202	0.254	0.98 dB
Grove 3	0.618	0.683	0.43 dB
Hydrangea	0.230	0.230	0.00 dB
Rubber Whale	0.161	0.161	0.00 dB
Urban 2	0.418	0.472	0.53 dB
Urban 3	0.662	0.897	1.32 dB
Venus	0.315	0.330	0.20 dB

As shown in Table 7, the new energy minimization framework results in an improvement of 0.43 dB when averaged over all of the sequences. For the sequences in Table 7 where no improvement was reported, the energy framework of (5.1) does a sufficient job of minimizing the block overlap through the use of the SAD and smoothness constraints, i.e., it is not necessary to incorporate an overlap regularizer. For the "Grove 2" sequence, the large improvement can be attributed to the ability of the overlap regularizer to reduce

errors around the occluded edges of the leaves. The large improvement for the "Urban 3" sequence is a result of the improvement in MVs around the edges of the image; the overlap regularizer prevents a large overlap of MC blocks at the edges. For a small region from the "Grove 2" sequence, we show the visual improvement of (5.6) over that of (5.1) using the motion-compensated frames in Fig. 5.5(a) and Fig. 5.5(b), respectively.

(a) Section of MC frame using (5.6).

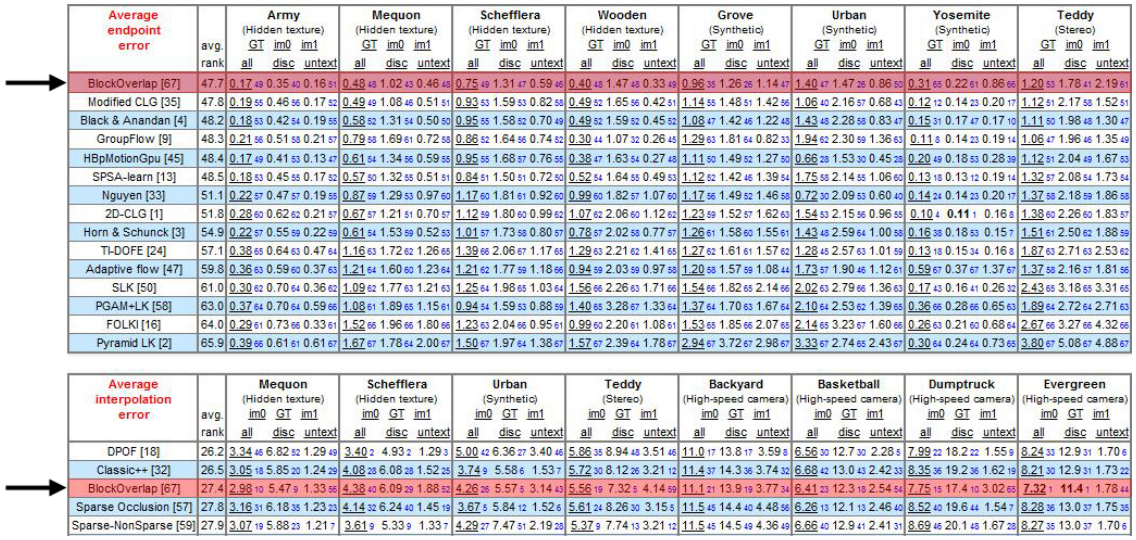
(b) Section of MC frame using (5.1).

Figure 5.5. Visual comparison of MC frames for "Grove 2" sequence.

In Fig. 5.5, improvements generated by (5.6) can be seen in the top half of the images near the left edges of the leaves. Similar improvements throughout the "Grove 2" sequence

are responsible for the large improvement reported in Table 7.

In addition to the results shown in Table 7, we also submitted our results to the Middlebury online benchmark for comparison with other motion estimation algorithms. The online results are shown in Fig. 5.6. In Fig. 5.6, our algorithm is shown to outperform several others in terms of endpoint and interpolation error. For the interpolation error, our algorithm outperforms all other block-based methods as well as several complex optical flow methods with long run times, and it produces the smallest interpolation error for the "Evergreen" sequence compared to all 69 algorithms currently in the database. In addition, it is the fastest non-GPU implementation at the time of this writing.



Average endpoint error	avg. rank	Army (Hidden texture) GT im0 im1 all	disc	untext	Mequon (Hidden texture) GT im0 im1 all	disc	untext	Schefflera (Hidden texture) GT im0 im1 all	disc	untext	Wooden (Hidden texture) GT im0 im1 all	disc	untext	Grove (Synthetic) GT im0 im1 all	disc	untext	Urban (Synthetic) GT im0 im1 all	disc	untext	Yosemite (Synthetic) GT im0 im1 all	disc	untext	Teddy (Stereo) GT im0 im1 all	disc	untext
BlockOverlap [67]	47.7	0.17 49	0.35 40	0.16 51	0.48 48	1.02 43	0.46 48	0.75 49	1.31 47	0.59 46	0.40 48	1.47 48	0.33 49	0.96 35	1.26 26	1.14 47	1.40 47	1.47 26	0.86 50	0.31 55	0.22 51	0.86 66	1.20 53	1.78 41	2.19 61
Modified CLG [35]	47.8	0.19 55	0.46 56	0.17 52	0.49 49	1.08 46	0.51 51	0.93 53	1.59 53	0.82 58	0.49 52	1.65 56	0.42 51	1.14 55	1.48 51	1.42 56	1.06 40	2.16 57	0.68 43	0.12 12	0.14 23	0.20 17	1.12 51	2.17 58	1.52 51
Black & Anandan [4]	48.2	0.18 53	0.42 54	0.19 55	0.58 52	1.31 54	0.50 50	0.95 55	1.58 52	0.70 49	0.49 52	1.59 52	0.45 52	1.08 47	1.42 46	1.22 48	1.43 48	2.28 58	0.83 47	0.15 31	0.17 47	0.17 10	1.11 50	1.98 48	1.30 47
GroupFlow [9]	48.3	0.21 56	0.51 58	0.21 57	0.79 58	1.69 61	0.72 58	0.86 52	1.64 56	0.74 52	0.30 44	1.07 32	0.26 45	1.29 63	1.81 64	0.82 33	1.94 62	2.30 59	1.36 63	0.11 8	0.14 23	0.19 14	1.06 47	1.96 45	1.35 49
HBpMotionGpu [45]	48.4	0.17 49	0.41 53	0.13 47	0.61 54	1.34 56	0.59 55	0.95 55	1.68 57	0.76 55	0.38 47	1.63 54	0.27 48	1.11 50	1.49 52	1.27 50	0.66 28	1.53 30	0.45 28	0.20 49	0.18 53	0.28 39	1.12 51	2.04 49	1.67 53
SPSA-learn [13]	48.5	0.18 53	0.45 55	0.17 52	0.57 50	1.32 55	0.51 51	0.84 51	1.50 51	0.72 50	0.52 54	1.64 55	0.49 53	1.12 52	1.42 46	1.39 54	1.75 58	2.14 55	1.06 60	0.13 18	0.13 12	0.19 14	1.32 57	2.08 54	1.73 54
Nguyen [33]	51.1	0.22 57	0.47 57	0.19 55	0.87 59	1.29 53	0.97 60	1.17 60	1.81 61	0.92 60	0.99 60	1.82 57	1.07 60	1.17 56	1.49 52	1.46 58	0.72 30	2.09 53	0.60 40	0.14 24	0.14 23	0.20 17	1.37 58	2.18 59	1.86 58
2D-CLG [1]	51.8	0.28 60	0.62 62	0.21 57	0.67 57	1.21 51	0.70 57	1.12 59	1.80 60	0.99 62	1.07 62	2.06 60	1.12 62	1.23 59	1.52 57	1.62 63	1.54 53	2.15 56	0.96 55	0.10 4	**0.11** 1	0.16 8	1.38 60	2.26 60	1.83 57
Horn & Schunck [3]	54.9	0.22 57	0.55 59	0.22 59	0.61 54	1.53 59	0.52 53	1.01 57	1.73 58	0.80 57	0.78 57	2.02 58	0.77 57	1.26 61	1.58 60	1.55 61	1.43 48	2.59 64	1.00 58	0.16 38	0.18 53	0.15 7	1.51 61	2.50 62	1.88 59
TI-DOFE [24]	57.1	0.38 65	0.64 63	0.47 64	1.16 63	1.72 62	1.26 65	1.39 66	2.06 67	1.17 65	1.29 63	2.21 62	1.41 65	1.27 62	1.61 61	1.57 62	1.28 45	2.57 63	1.01 59	0.13 18	0.15 34	0.16 8	1.87 63	2.71 63	2.53 62
Adaptive flow [47]	59.8	0.36 63	0.59 63	0.37 63	1.21 64	1.60 60	1.23 64	1.21 62	1.77 59	1.18 66	0.94 59	2.03 59	0.97 58	1.20 58	1.57 59	1.08 44	1.73 57	1.90 46	1.12 61	0.59 67	0.37 67	1.37 67	1.37 58	2.16 57	1.81 66
SLK [50]	61.0	0.30 62	0.70 64	0.36 62	1.09 62	1.77 63	1.21 63	1.25 64	1.98 66	1.03 64	1.56 66	2.26 63	1.71 66	1.54 66	1.82 65	2.14 66	2.02 63	2.79 66	1.36 63	0.17 43	0.16 41	0.26 32	2.43 65	3.18 65	3.31 65
PGAM+LK [58]	63.0	0.37 64	0.70 64	0.59 66	1.08 61	1.89 65	1.15 61	0.94 54	1.59 53	0.88 59	1.40 65	3.28 67	1.33 64	1.37 64	1.70 63	1.67 64	2.10 64	2.53 62	1.39 65	0.36 66	0.28 66	0.65 63	1.89 64	2.72 64	2.71 63
FOLKI [16]	64.0	0.29 61	0.73 66	0.33 61	1.52 66	1.96 66	1.80 66	1.23 63	2.04 66	0.95 61	0.99 60	2.20 61	1.08 61	1.53 65	1.85 66	2.07 65	2.14 65	3.23 67	1.60 66	0.26 63	0.21 60	0.68 64	2.67 66	3.27 66	4.32 66
Pyramid LK [2]	65.9	0.39 66	0.61 61	0.61 67	1.67 67	1.78 64	2.00 67	1.50 67	1.97 64	1.38 67	1.57 67	2.39 64	1.78 67	2.94 67	3.72 67	2.98 67	3.33 67	2.74 65	2.43 67	0.30 64	0.24 64	0.73 65	3.80 67	5.08 67	4.88 67

Average interpolation error	avg. rank	Mequon (Hidden texture) im0 GT im1 all	disc	untext	Schefflera (Hidden texture) im0 GT im1 all	disc	untext	Urban (Synthetic) im0 GT im1 all	disc	untext	Teddy (Stereo) im0 GT im1 all	disc	untext	Backyard (High-speed camera) im0 GT im1 all	disc	untext	Basketball (High-speed camera) im0 GT im1 all	disc	untext	Dumptruck (High-speed camera) im0 GT im1 all	disc	untext	Evergreen (High-speed camera) im0 GT im1 all	disc	untext
DPOF [18]	26.2	3.34 46	6.82 52	1.29 49	3.40 2	4.93 2	1.29 3	5.00 42	6.36 37	3.40 46	5.86 35	8.94 48	3.51 46	11.0 17	13.8 17	3.59 8	6.56 30	12.7 30	2.28 5	7.99 22	18.2 22	1.55 9	8.24 33	12.9 31	1.70 6
Classic++ [32]	26.5	3.05 18	5.85 20	1.24 29	4.08 28	6.08 28	1.52 28	3.74 9	5.58 6	1.53 7	5.72 30	8.12 26	3.21 12	11.4 37	14.3 36	3.74 32	6.68 42	13.0 43	2.42 33	8.35 36	19.2 36	1.62 19	8.21 30	12.9 31	1.73 22
BlockOverlap [67]	27.4	2.98 10	5.47 9	1.33 56	4.38 40	6.09 29	1.88 52	4.26 26	5.57 5	3.14 43	5.56 19	7.32 5	4.14 59	11.1 21	13.9 19	3.77 34	6.41 23	12.3 18	2.54 54	7.75 15	17.4 10	3.02 65	7.32 1	**11.4** 1	1.78 44
Sparse Occlusion [57]	27.8	3.16 31	6.18 38	1.23 23	4.14 32	6.24 40	1.45 19	3.67 5	5.84 12	1.52 6	5.61 24	8.26 30	3.15 5	11.5 45	14.4 40	4.48 56	6.26 13	12.1 13	2.46 40	8.52 40	19.6 44	1.54 7	8.28 36	13.0 37	1.75 35
Sparse-NonSparse [59]	27.9	3.07 19	5.88 23	1.21 7	3.61 9	5.33 9	1.33 7	4.29 27	7.47 51	2.19 28	5.37 9	7.74 13	3.21 12	11.5 45	14.5 49	4.36 49	6.66 40	12.9 41	2.41 31	8.69 45	20.1 48	1.67 28	8.27 35	13.0 37	1.70 6

Figure 5.6. Screenshots taken from Middlebury benchmark [3] with our endpoint error results (top) and interpolation error results (bottom) highlighted. The full tables are available at http://vision.middlebury.edu/flow/eval/results/.

5.8 Summary

The motion estimation framework presented in this chapter uses the block-overlap-based validity metric of Chapter 4 to improve the quality of the motion field. By introducing a block-overlap-based regularizer, we were able to provide a more uniform distribution of blocks, reduce the dependence on block size, and improve the quality of the motion field in

terms of endpoint error and interpolation error. The published results for the Middlebury sequences in Chapter 5.7 show that our method performs well compared to other state-of-the-art methods in the literature, and it is well-suited for applications which require a real-time approach.

CHAPTER 6

RELATED APPLICATION – CAMERA MISALIGNMENT CORRECTION FOR DEPTH ESTIMATION

In this chapter, we present a misalignment correction method for reducing depth errors that result from camera shift. The proposed method uses a real-time motion estimation approach to correct for alignment errors between stereo cameras. Unlike existing methods in the literature, the natural disparity between stereo views is incorporated into a constrained motion estimation framework. The proposed method is shown to accurately estimate synthetic misalignments due to translation, rotation, scaling, and perspective transformation. In addition, real images taken from a stereo camera rig confirm that the proposed method is capable of significantly reducing misalignments due to camera yaw, pitch, and roll.

6.1 Background

Stereo vision continues to grow in importance as more uses of 3-D technology emerge. Applications such as 3-D displays, cameras, robot navigation, and driver assistance systems take advantage of the natural disparity between two cameras to provide an estimate of 3-D parameters. However, the misalignment or mismatch between stereo cameras has been shown to affect 3-D depth estimates [90]. To correct for the misalignment or mismatch between left and right views, offline camera calibration is performed during the manufacturing process. However, in handheld, automotive, or robotic applications, cameras are often subject to environmental factors such as mechanical stresses, vibrations, or large temperature variations. These factors cause calibration parameters to drift, which can significantly affect the accuracy of the 3-D measurements [90][91]. Therefore, to maintain correspondence between left and right views, it is necessary to provide an online approach to misalignment correction.

It is important to distinguish misalignment correction from continuous self-calibration

methods, which were discussed in [91]. While the goal of self-calibration is to determine the extrinsic and intrinsic camera parameters, misalignment correction aims to reduce the mismatch between camera views without regard to the camera parameters. To determine the mismatch between views, it is necessary to determine the transformation model that maps pixels from the left to right image or vice versa. Such a transformation model should accommodate for multiple sources of error such a baseline shift, rotation, pitch, yaw, non-parallel sensors, and lens distortion. Unfortunately, a model which includes all error sources is prohibitively expensive and not suitable for a real-time approach.

To enable real-time misalignment correction, we make two assumptions regarding the calibration/alignment: 1) the cameras are initially calibrated to within a small error tolerance, and 2) the camera drift/misalignment results in small displacements between left and right images. The first assumption is necessary in order to determine an initial estimate of the horizontal disparity between left and right images. Otherwise, it is not possible to distinguish the alignment error from the natural horizontal disparity that exists between images. The second assumption allows us to use a block-based model to estimate pixel mappings between images, i.e., we approximate the different sources of alignment error as vertical and horizontal translations.

Other misalignment approaches have used an affine transformation model [92] and scale-invariant feature transform (SIFT) [93] to establish correspondence between left and right images. In [92], the authors remove the horizontal shift component from the affine model, which is equivalent to assuming that there is no camera misalignment due to horizontal shift, yaw, or non-parallel sensors. However, as shown in [90], depth estimates are most sensitive to camera motion which results in a relative yaw between views. In [93], the authors do not provide the details of their SIFT approach or experimental data, but such approaches generally require a large number of feature points and a method for outlier detection.

In the next sections, we examine how camera alignment errors affect depth estimates

and propose a misalignment correction scheme based on constrained motion estimation using block matching. In Chapter 6.2, we summarize the works of [90][91], which present both theoretical and experimental results for the effects of alignment errors on depth estimates. In Chapter 6.3.2, we use synthetically-generated images to show that the our block-based motion estimation algorithm is capable of detecting various alignment errors. The proposed misalignment correction scheme is presented in Chapter 6.4. Experimental results from a stereo camera rig are presented in Chapter 6.5, and a summary is given in Chapter 6.6.

6.2 Effect of Alignment Errors

In this section, we summarize the results of several works that provide both the equations and experimental data relating alignment errors to 3-D depth errors [90][91].

The different types of alignment error for stereo cameras (with the exception of image sensor misalignment and scaling) are shown in Fig. 6.1.

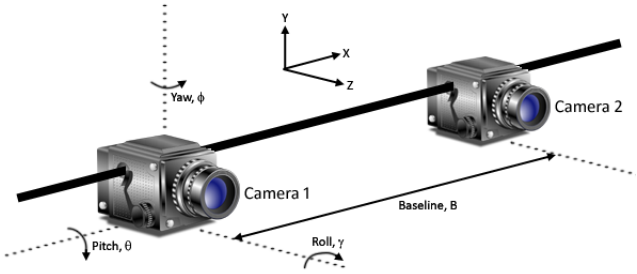


Figure 6.1. Stereo camera rig showing error sources.

As shown in Fig. 6.1, depth errors may result from baseline shift, rotation (roll), pitch, and yaw. In [90], the authors found that the most critical alignment parameters in order of importance were yaw, image sensor tilt, pitch, roll, baseline error, and focal length error. The equations relating the sensitivity of these parameters to depth are given in Table 8.

Table 8. Relative changes in depth for different misalignments

Error Source	Relative Change in Depth
Yaw Error $\Delta\phi$	$\frac{\Delta Z}{\Delta\phi} \approx -\frac{Z^2}{B}(1 + X_2)$
Sensor Tilt $\Delta\phi$	$\frac{\Delta Z}{\Delta\phi} \approx -\frac{X_2^2}{B}$
Pitch Error $\Delta\theta$	$\frac{\Delta Z}{\Delta\theta} \approx \frac{Z^2}{B}X_2Y_2$
Roll Error $\Delta\gamma$	$\frac{\Delta Z}{\Delta\gamma} \approx \frac{Z^2}{B}Y_2$
Baseline Error ΔB	$\frac{\Delta Z}{\Delta B} \approx -\frac{Z}{B}$
Focal Length Error Δf_2	$\frac{\Delta Z}{\Delta f_2} \approx -\frac{Z^2}{Bf_2^2}$

To generate the equations in Table 8, it is assumed that Camera 1 is perfectly calibrated, and Camera 2 is perfectly calibrated except for the error source listed. The error sources in the left column of the table are the deviations from perfect alignment with respect to camera 2, where X_2, Y_2 represents the true 3-D coordinate of the object with respect to camera 2, B is the baseline distance, Z is the true absolute depth, and f_2 is the focal length. The error sources which contribute the most to alignment errors can be explained by the order of the depth Z in the right column; for example, the depth error increases quadratically for yaw misalignment but decreases linearly for baseline misalignment.

The results in Table 8 show that in order to minimize depth errors, the yaw, pitch, and sensor tilt cannot be ignored. These errors will introduce a prospective transformation between camera views since the 2-D displacement of objects will depend on their depth,

and this type of transformation cannot handled by the affine model of [92].

It should also be noted that the correction of small misalignments may not reduce the depth error. As discussed in [94], the spatial quantization of the image plane places an upper bound on the depth error that can be improved by calibration or misalignment correction.

6.3 Block-Based Detection of Alignment Errors

In this section, we demonstrate that our block-based motion estimation algorithm is capable of detecting alignment errors with a high degree of accuracy.

6.3.1 Overview of Block-Based Algorithm

Our algorithm performs true motion estimation using quarter-pixel motion vectors (MVs), and it has achieved high rankings compared to other state-of-the-art algorithms in the Middlebury database [95] (see "BlockOverlap" in database). The Middlebury database contains image sequences with various types of motion: translation, rotation, scaling, perspective, etc.

As part of the motion estimation process, our algorithm assigns a confidence value to each MV. The confidence value is used to characterize the validity of the MV, i.e., a higher confidence value indicates that the chosen MV is more likely to represent the true motion. More details can be found in Chapter 4.

Assigning a confidence value to each MV is important for real stereo images, where there may be occlusions, brightness variations, etc. In such cases, the MVs will have low confidence values and thus cannot be used to estimate the amount of misalignment. These MVs must be handled separately when estimating the depth; however, this topic is outside the scope of our work. In order to use motion to estimate the horizontal disparity in Chapter 6.4, we assume that misalignment correction is performed on scenes that do not contain independent object motion. Although this may somewhat limit the proposed misalignment correction approach, it is much less limiting than the need to perform manual calibration.

6.3.2 Detecting Alignment Errors

We now show that our block-based motion estimation algorithm is capable of estimating different types of misalignments on synthetic sequences. We use a global image translation to model baseline shift between cameras, rotation to model roll, scaling to model focal length differences, and perspective transformation to model yaw, pitch, and sensor tilt.

The 300-frame "Foreman" sequence was used to test each type of misalignment error. For each frame of the sequence, we created a synthetic frame with the specified misalignment, and we used our motion estimation algorithm to estimate the amount of misalignment. For the translational, scaling, and rotational misalignments, we compare our estimated misalignment to that of [92].

6.3.2.1 Translation

The translational misalignment was created by shifting all pixels in each frame by the specified amount. The motion was then estimated between the shifted and original frames. A comparison between our algorithm and [92] is shown in Table 9 for three different vertical shift amounts.

Table 9. Comparison of errors for translational misalignment.

Translation Params.	**Method of [92]**			**Proposed Method**		
True Shift	-10	24	48	-10	24	48
Estimated Shift	-10.17	24.55	46.97	-10.00	24.00	48.00
Mean Error	0.17	0.55	1.03	0.00	0.00	0.00
Standard Deviation	0.28	0.45	0.79	0.00	0.00	0.00

6.3.2.2 *Rotation*

The rotational misalignment was created by multiplying each pixel position in the original frame by a rotation matrix, i.e.,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\gamma & \sin\gamma & \frac{W}{2}(1-\cos\gamma) - \frac{H}{2}\sin\gamma \\ -\sin\gamma & \cos\gamma & \frac{W}{2}\sin\gamma + \frac{H}{2}(1-\cos\gamma) \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{6.1}$$

where $\{x', y'\}$ are the mapped pixel coordinates, $\{x, y\}$ are the original pixel coordinates, γ is the rotation angle, and W, H are the frame width and height, respectively.

For non-integer positions of x' and y' in (6.1), bilinear interpolation was used to estimate the pixel intensities. Next, the motion was estimated between the rotated and original frames. The angle of rotation was found from the MVs using the arc length approximation

$$\gamma_d = \frac{s}{r}, \tag{6.2}$$

where γ_d is the change in rotation angle, s is the displacement of a given pixel, and r is the distance from the pixel to the center of the frame. The rotation angle was estimated for each pixel $\{x', y'\}$ in the rotated frame. A comparison between our algorithm and [92] is shown in Table 10.

Table 10. Comparison of errors for rotational misalignment.

Rotation Params.	**Method of [92]**			**Proposed Method**		
True Angle	$-3°$	$4°$	$7°$	$-3°$	$4°$	$7°$
Estimated Angle	-2.95	3.95	5.66	-2.99	3.99	6.98
Mean Error	0.05	0.05	1.34	0.004	0.007	0.02
Standard Deviation	0.22	0.23	0.48	0.002	0.002	0.005

6.3.2.3 *Scaling*

The scaling misalignment was created by multiplying each pixel position in the original frame by a scaling matrix, i.e.,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 & \frac{W}{2}(1-s_x) \\ 0 & s_y & \frac{H}{2}(1-s_y) \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{6.3}$$

where $\{x', y'\}$ are the mapped pixel coordinates, $\{x, y\}$ are the original pixel coordinates, the scaling factor $s_x = s_y$, and W, H are the frame width and height, respectively.

For non-integer positions of x' and y' in (6.3), bilinear interpolation was used to estimate the pixel intensities. Next, the motion was estimated between the scaled and original frames. The scaling factor was computed by comparing the ratio of displacements in the original and scaled frames as follows:

$$s_x = s_y = \frac{\sqrt{(x-\frac{W}{2})^2 + (y-\frac{H}{2})^2}}{\sqrt{(j_x-\frac{W}{2})^2 + (i_y-\frac{H}{2})^2}}, \tag{6.4}$$

where $\{x, y\}$ are the pixel coordinates in the original frame, $\{i_y, j_x\}$ are the motion-compensated pixel coordinates in the scaled frame, and W, H are the frame width and height, respectively. The scaling factor was estimated for each pixel in the scaled frame. A comparison between our algorithm and [92] is shown in Table 11.

Table 11. Comparison of errors for scaling misalignment.

Scaling Params.	**Method of [92]**			**Proposed Method**		
True Scale	-3	4	5	-3	4	5
Estimated Scale	-2.83	3.66	3.75	-2.99	3.99	4.99
Mean Error	0.17	0.34	1.25	0.007	0.007	0.005
Standard Deviation	0.35	0.32	0.30	< 0.00	< 0.00	< 0.00

6.3.2.4 *Perspective Transformation*

A perspective transformation can be used to model yaw, pitch, and roll (rotation) misalignments. Since rotation was covered in Chapter 6.3.2.2, we focus on yaw and pitch misalignments in this section.

To generate frames with yaw and pitch misalignments, it is necessary to represent 3-D coordinates in a 2-D plane such that an object has a smaller projection when it is far away from the center of projection and a larger projection when it is closer. The 3-D perspective mapping is given in (6.5), which is a modification of the 3-D rotation matrix in x-y-z convention for Euler angles of θ, γ, and ϕ (pitch, roll, and yaw, respectively).

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\gamma & -\cos\phi\sin\gamma + \sin\phi\sin\theta\cos\gamma & (\sin\phi\sin\gamma + \cos\phi\sin\theta\cos\gamma)f \\ \cos\theta\sin\gamma & \cos\phi\cos\gamma + \sin\phi\sin\theta\sin\gamma & (-\sin\phi\cos\gamma + \cos\phi\sin\theta\sin\gamma)f \\ -\dfrac{\tan\theta}{f\cos\phi} & \dfrac{\tan\phi}{f} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{6.5}$$

The general 3-D rotation matrix was modified in order to provide realistic values for the focal length, f, and the angle of view, α. The focal length and angle of view are related as follows:

$$\alpha = 2\arctan\frac{d}{2f}, \tag{6.6}$$

where d is the size of the image sensor in the horizontal or vertical direction. We chose $\alpha = 60^\circ$ and $d = 16mm$ for a more realistic depiction of the different 3-D rotations.

In order to display the 3-D coordinates in a 2-D plane, the coordinates of $\{x', y'\}$ in (6.5) must be normalized by the depth component, z'. The 2-D mapping equations are given as

$$\begin{aligned} x'' &= \frac{x'}{z'} - l_x \\ y'' &= \frac{y'}{z'} - l_y, \end{aligned} \tag{6.7}$$

where l_x and l_y are used to shift the projected frame such that the center of projection is located at $(W/2, H/2)$, and W and H are the frame width and height, respectively.

For non-integer positions of x'' and y'' in (6.7), bilinear interpolation was used to estimate the pixel intensities. Next, the motion was estimated between the projected and original frames. To determine the yaw, pitch, and roll parameters, it is first necessary to estimate the original mapping matrix used to map the original image coordinates onto the projected frame. We re-write the linear system of (6.5) as

$$\mathbf{x}' = \mathbf{H}\mathbf{x}, \tag{6.8}$$

where $\mathbf{x}'$ represents the projected frame coordinates, $\mathbf{H}$ is the modified rotation matrix, and $\mathbf{x}$ represents the original frame coordinates.

The modified rotation matrix can be found by determining the plane-to-plane homography $\hat{\mathbf{H}}$ between the original and projected frames, where $\hat{\mathbf{H}}$ is an estimate of $\mathbf{H}$. Given four sets of pixel coordinates in the projected frame and the corresponding four sets of motion-compensated pixel coordinates in the original frame[1], an estimate of $\hat{\mathbf{H}}$ can be determined. More details can be found in [96]. Following the estimation of $\hat{\mathbf{H}}$, the angles of the yaw, pitch, and roll can be determined by equating the entries of $\hat{\mathbf{H}}$ with the original $\mathbf{H}$ given in (6.5).

For small angles ($< 10^\circ$) of yaw, pitch, and roll, pixels near the center of projection will undergo very small displacements in the projected frame. As a result, the estimate of $\hat{\mathbf{H}}$ becomes ill-conditioned near the center of projection. To overcome this limitation, we manually selected four pixel locations near the image corners in the projected frame and determined their corresponding motion-compensated pixels in the original frame. An illustration of the pixel selection is shown in Fig. 6.2.

[1]The motion-compensated pixel coordinates must be shifted by l_x and l_y prior to estimating $\hat{\mathbf{H}}$

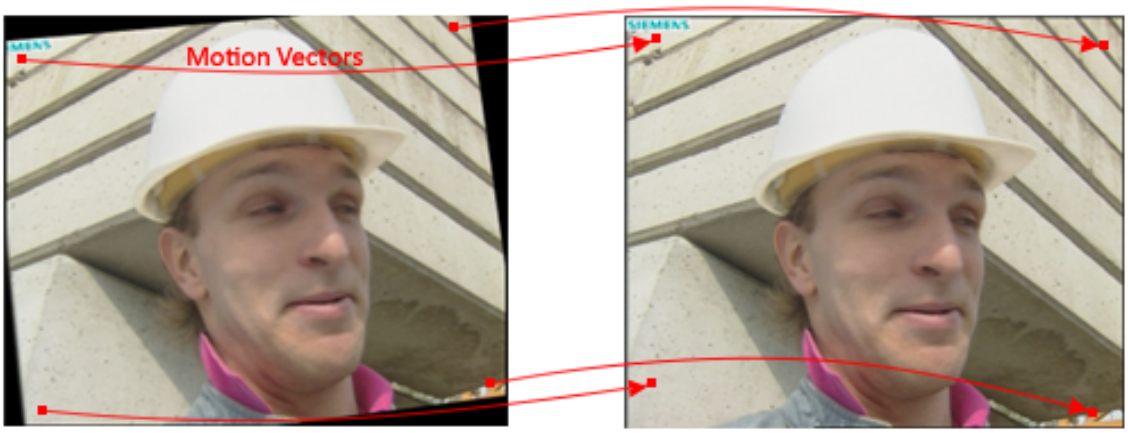


Figure 6.2. Pixel selection in original and projected frames.

Since perspective transformations were not handled by the method of [92], only the results of our algorithm are shown in Table 12.

Table 12. Errors for yaw ϕ and pitch θ misalignments.

Params.	ϕ	ϕ	ϕ	θ	θ	θ
	−3°	**4°**	**7°**	**−3°**	**4°**	**7°**
Estimated Angle	-2.30	4.68	7.31	-2.93	4.15	7.06
Mean Error	0.72	0.68	0.31	0.07	0.15	0.06
Standard Deviation	0.52	0.22	0.19	0.10	0.08	0.30

The larger errors for the small angles in Table 12 are mainly due to the difficulty in estimating $\hat{\mathbf{H}}$ for small angles. In order to estimate small angles accurately, very high precision motion vectors are needed; however, our algorithm only computes quarter-pixel accurate MVs.

Since real stereo cameras generally suffer from multiple misalignments, we also generated a projected frame with all three types of misalignments – yaw, pitch, and roll. As described in the previous experiments, the matrix $\hat{\mathbf{H}}$ was estimated using four sets of pixel coordinates in the original and projected frame. The results for the three simultaneous

misalignments are shown in Table 13.

Table 13. Errors for yaw ϕ, pitch θ, and roll γ misalignments.

Params.	ϕ	θ	γ
	7°	**7**°	**7**°
Estimated Angle	7.14	7.17	6.98
Mean Error	0.14	0.17	0.18
Standard Deviation	0.46	0.27	0.48

The results for the synthetically-generated sequences of Table 9 - Table 13 show that our algorithm is capable of detecting different types of misalignments with high accuracy.

6.4 Proposed Misalignment Correction

In this section, we first discuss the constrained motion estimation model which will be used to preserve the stereo disparity and estimate alignment errors. The constrained motion model will then be demonstrated on real images from a stereo camera rig.

The constrained motion model was introduced by Farsiu et al. [97] in the context of super-resolution. In [97], the authors introduced the Jacobi identity for three consecutive frames i, j, and k as follows:

$$V_{i,k} = V_{i,j} + V_{j,k}, \tag{6.9}$$

where $V_{i,k}$ is the motion vector between frames i and k. Equation (6.9) states that motion between frames i and k must be the composition of the motion between frames i, j and j, k. We note that (6.9) assumes a translational motion model, which was demonstrated to be an effective means of estimating alignment errors in Chapter 6.3.2.

We can extend (6.9) to misalignment correction by decomposing the misalignment problem into two steps: 1) determine the initial horizontal shifts $\underline{X}_D$ between right and

left views[2], and 2) determine the horizontal and vertical shifts between right and left views resulting from misalignment, $\underline{X}_M$. The vectors $\underline{X}_D$ and $\underline{X}_M$ represent a lexicographic ordering for the estimated motion with respect to the right view. The misalignment error can then be written as

$$\underline{E}_M = \underline{X}_M - \underline{X}_D, \tag{6.10}$$

where $\underline{E}_M$ is a vector of misalignment errors for each pixel in the right view. We now examine each of the above steps in detail.

Step 1): To determine the initial horizontal shifts between right and left views, we assume that the two cameras have been pre-calibrated. It is important that the initial calibration is accurate since it will serve as a baseline for the misalignment correction algorithm. Since calibration is already performed during the stereo camera manufacturing process, no further overhead is incurred by requiring an initial calibration. The initial calibration produces the intrinsic and extrinsic cameras parameters which are used to remove lens distortion and provide image rectification. Following distortion removal and rectification, the left and right image planes will be vertically aligned. Next, the baseline horizontal shifts $\underline{X}_D$ can be determined by estimating the horizontal motion between right and left views. It is only necessary to determine $\underline{X}_D$ once since it can be stored in memory for future access.

Step 2): If the cameras remain calibrated, it is not necessary to perform the second step; i.e., $\underline{E}_M = \underline{0}$. However, real cameras will shift due to environmental factors that introduce horizontal and/or vertical displacements between the right and left views. Given two new images taken from misaligned cameras, distortion removal and rectification is first performed using the initial calibration parameters. However, since the initial calibration data no longer applies to the new images, there will exist both horizontal and vertical displacements between rectified views. The motion between rectified views is represented by $\underline{X}_M$. The alignment error can then be computed using (6.10).

[2]The left or right view may be chosen as the reference frame for motion estimation.

6.5 Results from Stereo Camera Rig

To demonstrate the proposed misalignment correction method from Chapter 6.4, we used seven stereo image pairs taken from a camera rig similar to that of Fig. 6.1. Only alignment errors due to yaw, pitch, and roll were considered, i.e., it is assumed that the baseline B does not change between cameras, and the differences in focal length are sufficiently small.

For the first stereo image pair, which will be used as the baseline for determining $\underline{X}_D$, we roughly aligned the left and right cameras such that the optical axes were parallel to one another. To perform the calibration of the two cameras, we used the OpenCV implementation of [98][99]. The method of [99] is a manual calibration method which requires multiple images of a chessboard pattern in different orientations. We performed three separate calibrations, acquiring 30 chessboard images in each calibration. The intrinsic and extrinsic camera parameters for the three calibrations were averaged and taken to be the initial parameters. The relative yaw, pitch, and roll (ϕ, θ, γ) from the rotation matrix were found to be

$$\begin{bmatrix} \phi \\ \theta \\ \gamma \end{bmatrix} = \begin{bmatrix} 3.93^\circ \\ 0.63^\circ \\ -0.57^\circ \end{bmatrix}. \tag{6.11}$$

Following the determination of the calibration parameters, we again used the OpenCV implementation of [98][99] to perform image rectification and distortion removal. The rectified images for the left and right views are shown in Fig. 6.3(a) and Fig. 6.3(b), respectively.

(a) Left rectified image. (b) Right rectified image.

Figure 6.3. Left and right rectified images for initial calibration parameters.

To generate the remaining six stereo image pairs, the yaw, pitch, or roll of the right camera was modified while the position of the left camera remained unchanged. After each adjustment to the right camera, three calibrations were performed to determine the relative rotation matrix. The relative yaw, pitch, and roll for the six stereo image pairs are given in Table 14.

Table 14. Relative yaw ϕ, pitch θ, and roll γ from calibration data.

Experiment #	ϕ	θ	γ
2	−2.03°	1.76°	−0.54°
3	6.46°	1.73°	−0.40°
4	3.10°	−2.70°	−0.03°
5	2.24°	2.21°	−0.17°
6	3.61°	2.10°	1.28°
7	1.42°	2.35°	−1.89°

For Experiments #2 and #3 in Table 14, the yaw ϕ of the right camera was modified in both directions while the left camera remained unchanged. Similarly, the pitch θ was modified in both directions in Experiments #4 and #5, and the roll γ was modified in both

directions in Experiments #6 and #7. Since our camera rig was imperfect, it did not allow us to completely isolate each angle individually (e.g., changing the pitch angle resulted in small deviations in yaw and roll angles).

For each of the experiments shown in Table 14, the captured left and right images were rectified using the initial calibration parameters. Next, the motion was estimated between the right and left images to determine $\underline{X}_M$. Since $\underline{X}_D$ was previously determined from the initial calibration parameters, the translational alignment error $\underline{E}_M$ was found from (6.10).

Using the alignment error $\underline{E}_M$, the pixel positions in the right image were motion-compensated to determine if alignment errors had been reduced. To estimate the alignment errors, we compare motion-compensated pixel positions in the right image to the initial right image (the baseline). As in Chapter 6.3.2.4, the point mapping method was used to determine the matrix that maps the four sets of points in the initial right image to the motion-compensated points. These points were manually selected in the initial images, and the corresponding motion-compensated points were determined. We only considered pixels whose MVs had high confidence values (see Chapter 6.3.1). The angular errors for the yaw, pitch, and roll between the initial pixels and motion-compensated pixels are given in Table 15.

Table 15. Angular differences in yaw ϕ, pitch θ, and roll γ.

Experiment #	ϕ	θ	γ
2	$< 0.01°$	$-0.55°$	$< 0.01°$
3	$< 0.01°$	$1.48°$	$< 0.01°$
4	$< 0.01°$	$1.25°$	$< 0.01°$
5	$< 0.01°$	$0.20°$	$< 0.01°$
6	$< 0.01°$	$-0.55°$	$< 0.01°$
7	$< 0.01°$	$-1.37°$	$< 0.01°$

As shown in Table 15, the proposed misalignment correction method significantly reduces the misalignment due to yaw and roll, and it reduces the pitch misalignment to within 1.5°.

6.6 Summary

The misalignment correction approach proposed in this chapter uses a block-matching-based constrained motion framework to preserve the natural disparity between stereo views while correcting for vertical and horizontal misalignments that result from camera shift. The proposed approach requires only an initial calibration to determine the natural disparity, and the vertical and horizontal misalignments are estimated using real-time block-based motion estimation. The ability of the block-based motion estimation algorithm to correct for misalignments was first demonstrated using synthetic sequences with translation, rotation, scaling, and perspective transformations in Chapter 6.3. In Chapter 6.5, it was shown that proposed algorithm is capable of reducing all types of misalignments to within 1.5°. For the yaw angle misalignment, which has the greatest effect on depth estimates, the proposed algorithm reduces the misalignment error to less than 0.01°.

CHAPTER 7

CONCLUSION AND FUTURE RESEARCH DIRECTIONS

Before summarizing our contributions, we first look at the complexity of the motion estimation algorithm and POCS algorithm in order to demonstrate the reduced complexity of our SR algorithm. The complexity is given in terms of the operation counts required for each algorithm.

7.1 Motion Estimation Complexity

Let the original size of the images (level 2 in a three-level image pyramid) be given by MxN, where M is the width and N is the height of the image. Further, let the image be divided into blocks of size BxB with block matching search range RxR, where the size of B and R are given for the lowest resolution level (level 3) of the pyramid. We summarize these parameters in Table 16.

Table 16. Block matching parameters for complexity analysis.

Level #	Resolution	Block Size	Search Size
1	$2M \times 2N$	$\frac{B}{4} \times \frac{B}{4}$	$\frac{R}{4} \times \frac{R}{4}$
2	$M \times N$	$\frac{B}{2} \times \frac{B}{2}$	$\frac{R}{2} \times \frac{R}{2}$
3	$\frac{M}{2} \times \frac{N}{2}$	$B \times B$	$R \times R$

Using block matching only, we give the required number of additions and comparisons that are necessary for each level of the pyramid. The number of additions and comparisons include the number of blocks that must be searched, and the number of the SAD operations

for each block. The operation counts for block matching are given in Table 17.

Table 17. Number of additions and comparisons for block matching.

Level #	Additions	Comparisons
1	$8MN\left(\frac{R}{4}-\frac{B}{4}+1\right)^2$	$\frac{4MN}{B^2}\left(\frac{R}{4}-\frac{B}{4}+1\right)^4$
2	$2MN\left(\frac{R}{2}-\frac{B}{2}+1\right)^2$	$\frac{MN}{B^2}\left(\frac{R}{2}-\frac{B}{2}+1\right)^4$
3	$\frac{MN}{2}(R-B+1)^2$	$\frac{MN}{4B^2}(R-B+1)^4$

Next, we look at the number of operations required for regularization, which includes both the spatial regularizer (eight-connected neighbor MVs) and the block overlap regularizer. For each block in the image, there are nine spatial MVs tested. In addition, each tested MV requires a SAD calculation, smoothness penalty calculation, and overlap calculation. We consider the worst case scenario here; i.e., all spatial MVs are distinct. However, the number of operations will decrease if any of the spatial MVs are the same. For each level of the pyramid, three iterations are required in order for the MVs to converge, and we include the three iterations in the determination of the number of additions and comparisons required. The operation counts for regularization are given in Table 18.

Table 18. Number of additions and comparisons for regularization.

Level #	Additions	Comparisons
1	$\frac{1728MN}{B^2}\left(\frac{3B^2}{16}+24\right)$	$\frac{1728MN}{B^2}$
2	$\frac{108MN}{B^2}\left(\frac{3B^2}{4}+24\right)$	$\frac{108MN}{B^2}$
3	$\frac{27MN}{4B^2}\left(3B^2+24\right)$	$\frac{27MN}{4B^2}$

Note that the operation counts in Table 18 have to be computed for different block sizes at each level of the pyramid. For example, if the initial block size at level 3 of the pyramid is 32x32, then the total number of regularization operations for level 3 can be found by adding up the number of operations for block sizes of 32x32, 16x16, 8x8, 4x4, and 2x2.

To get an idea of the total number of additions and comparisons involved for block matching and regularization, we consider two images with VGA resolution (640x480), and the block and search sizes given in Chapter 3.5 (re-shown in Table 19 below).

Table 19. Block sizes and search sizes for three-level image pyramid

Level	Block Size	Search Size
Level 1	8	32
Level 2	16	64
Level 3	32	128

We consider the worst case scenario, i.e., there are no repeated MVs. The total number of additions and comparisons for the complete motion estimation algorithm given the above parameters is 9.35×10^9 and 9.02×10^9, respectively. The processor that was used in our

work is the Intel Core i7 875K, which is capable of doing integer additions and comparisons in one cycle or less (for pipelined instructions). The rated MIPS for the i7 875K processor is approximately 92,100 MIPS, which means that the entire motion estimation algorithm at VGA resolution takes approximately 0.2 seconds in the worst-case scenario. Actual run-times for the algorithm typically vary between 0.1 - 0.15 seconds since there are generally a large number of duplicate MVs.

To compare the performance of our motion estimation algorithm to other state-of-the-art motion estimation algorithms, we have sorted the results from the Middlebury benchmark [3] based on the average run-time of the "Urban 3" VGA sequence. The interpolation error and endpoint error sorted by performance is shown in Table 20.

Table 20. Comparison of interpolation error and endpoint error for different motion estimation algorithms sorted by run time.

Algorithm	Run Time (seconds)	Interpolation Error Rank	Endpoint Error Rank	GPU
[23] Rannacher	0.12	48.1	35.8	Yes
[43] Bartels	0.15	34.4	39.1	Yes
[58] PGAM+LK	0.37	55.3	65	No
[36] ComplOF-FED-GPU	0.97	24.5	27.4	Yes
[16] FOLKI	1.4	47.2	66	No
[52] SimpleFlow	1.7	48.4	21.2	No
[22] Aniso. Huber-L1	2	14.7	28.2	Yes
[67] BlockOverlap	2	28.2	49.6	No
[17] TV-L1-improved	2.9	38.8	32.2	Yes
[15] F-TV-L1	8	23.2	34.9	No

The "Interpolation Error Rank" and "Endpoint Error Rank" numbers given in Table 20 were determined by Middlebury University using a weighting of errors for eight different sequences. A lower rank value represents less error. Our motion estimation method is listed as "[67] BlockOverlap" in the table, and it is the 3rd best in terms of interpolation error and 12th best in terms of endpoint error. At the time of this writing, there are currently 69 algorithms in the online database. In the last column of Table 20, we also note whether the algorithms have been implemented on a GPU or a CPU. The run time of our algorithm in Table 20 (2 seconds) is based on quarter-pixel accurate MVs. The run times of 0.1 - 0.15 seconds are for half-pixel accurate MVs.

7.2 POCS Complexity

Since our SR algorithm is based on POCS method, it is not possible to directly calculate the number of operations required. The POCS method requires a different number of iterations depending on the amount of error between the original and projected pixels. Therefore, to get an idea of the complexity, we look at the run time required for the POCS method assuming that motion estimation has already been performed. The HR image was generated by applying the POCS method to the motion estimates from six LR images. We used five different VGA video sequences to estimate the average run time for the POCS algorithm, and we found the worst case run time to be approximately 0.29 seconds.

7.3 Overall Complexity

Given the motion estimation run times from Chapter 7.1 and the POCS run times from Chapter 7.2, we can approximate the run time for the complete SR algorithm. Based on the worst case run time from the motion estimation algorithm (0.2 seconds x 6 image pairs), and the worst case run time from the POCS algorithm (0.19 seconds), the run time of the SR algorithm for VGA video is approximately 1.39 seconds.

While a run time of 1.39 seconds is not suitable for a real-time approach (e.g., one that

requires a frame rate of 30 frames-per-second), it is worth noting that we have not made any performance optimizations, and the SR algorithm runs on a single thread.

7.4 Contributions

In this thesis, our focus has been to create a low-complexity motion estimation algorithm for use with super-resolution. We have shown that the proposed block-based motion-estimation algorithm rivals many of the more complex state-of-the-art optical-flow-based motion estimation algorithms. In addition, the proposed algorithm is one of the few state-of-the-art algorithms suitable for use with super-resolution.

Even the most complex motion estimation algorithms fail in the presence of occlusions and other complex motions. Therefore, any practical motion estimation algorithm must be capable of detecting when errors occur and preventing such errors from affecting the intended application. In this thesis, we have developed a novel block-based validity metric and shown its superior performance compared to other validity metrics in the literature.

In terms of the low-complexity motion estimation algorithm and validity metric developed in this thesis, we now discuss our specific contributions.

1. Block Matching Improvements - We have improved the quality of block matches by improving the search strategy in Chapter 3.2, and by incorporating prior motion vectors into the hierarchical block matching framework (Chapter 3.4).

2. Regularization Improvements - We have improved the quality of the motion field by choosing a robust penalty function and determining which motion vectors to include in the minimization framework (Chapter 3.3). In addition, we performed a sensitivity analysis on the Lagrange multiplier to determine an appropriate initialization to avoid oversmoothing the motion vectors (Chapter 3.3).

3. Motion Vector Validity - We developed a novel block-based validity metric and compared our method with several other validity metrics in the literature. Our method

was shown to be superior to all other validity methods for a wide range of image and video sequences (Chapter 4).

4. New regularizer - Based on our block-based validity method, we introduced a new regularizer into the energy minimization framework and showed that the new regularizer significantly improves the quality of the motion field (Chapter 5).

5. Camera Misalignment - We demonstrated that our motion estimation framework can be used to correct camera misalignments that are present in stereo cameras. Our method was shown to reduce various angular misalignments using a stereo camera rig (Chapter 6).

6. Super-Resolution with Validity Metric - We demonstrated that our validity metric can be used together with the projection onto convex sets (POCS) method in order to prevent significant artifacts from appearing in the reconstructed high-resolution (HR) image . Our validity metric was also compared to other validity metrics in the context of super-resolution, and it was shown to provide greater artifact reduction without sacrificing image quality (Chapter 4.3).

7.5 Future Research Directions

The motion estimation problem is far from solved, and unfortunately, there is not much emphasis on low-complexity approaches. However, some of the more complex approaches offer some insight into new ideas that may have low-resolution analogues. For example, the data similarity metric may be improved by taking advantage of color information, and new research shows that the normalized cross-correlation may be better-suited than the traditional sum of absolute differences (SAD) metric.

For the regularization term, more research is needed to find a low-complexity analogue of the total-variation norm, which is the predominant norm used in all complex optical flow approaches. In addition, other non-gradient based norms should be considered, especially

since the gradient is difficult to estimate and performs poorly in noise.

As the required subpixel-accuracy of motion vectors gets smaller (more decimal places), the computation time for interpolation-based motion estimation algorithms nearly doubles for every doubling of the image resolution. Although several approaches to sub-pixel motion estimation without interpolation have been proposed, none of them address the problem of generating subpixel motion vectors in the presence of occlusion. More research is needed to determine how to use the motion vector validity in conjunction with such approaches.

Finally, more research is needed to determine how to correct for the angular misalignments detected between stereo cameras. Since there is not a one-to-one mapping between pixels in stereo camera views, shifting one of the images to compensate for misalignment may reveal holes. In such cases, it is necessary to determine how to use the motion vectors and their corresponding validity to fill the holes.

REFERENCES

[1] A. Patti, M. Sezan, and A. Tekalp, "Robust methods for high-quality stills from interlaced video in the presence of dominant motion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 328 –342, Apr. 1997.

[2] D. Wang, A. Vincent, and P. Blanchfield, "Hybrid de-interlacing algorithm based on motion vector reliability," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, pp. 1019 – 1025, Aug. 2005.

[3] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, pp. 1–31, 2011.

[4] M. Kanellos, "New life for moore's law," *CNET News.com*, 2005.

[5] F. Lin, C. Fookes, V. Ch, and S. Sridharan, "Investigation into optical flow super-resolution for surveillance applications," 2005.

[6] F. Li, X. Jia, and D. Fraser, "Universal hmt based super resolution for remote sensing images," in *ICIP 2008. 15th IEEE International Conference on Image Processing*, pp. 333–336, Oct. 2008.

[7] F. Li, X. Jia, D. Fraser, and A. Lambert, "Super resolution for remote sensing images based on a universal hidden markov tree model," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, pp. 1270–1278, Mar. 2010.

[8] Z. Yan, Y. Lu, and H. Yan, "Reducing the spiral ct slice thickness using super resolution," in *2010 17th IEEE International Conference on Image Processing (ICIP)*, pp. 593–596, Sept. 2010.

[9] A. Gholipour, J. Estroff, and S. Warfield, "Robust super-resolution volume reconstruction from slice acquisitions: Application to fetal brain mri," *IEEE Transactions on Medical Imaging*, vol. 29, pp. 1739–1758, Oct. 2010.

[10] G. Chang, T. Pan, J. Clark, and O. Mawlawi, "Implementation and optimization of a new super-resolution technique in pet imaging," in *ISBI '09. IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, p. 253, July 2009.

[11] G. Clement, "Superresolution in ultrasound imaging," in *ISBI '09. IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pp. 258–261, July 2009.

[12] A. Patti, M. Sezan, and A. Tekalp, "High resolution standards conversion of low resolution video," in *ICASSP-95. International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, pp. 2197–2200 vol.4, May 1995.

[13] S. Baker and T. Kanade, "Limits on super-resolution and how to break them," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1167–1183, Sept. 2002.

[14] W. Zhang and W.-K. Cham, "High quality artifact-free super-resolution," in *ICIP2010. 17th IEEE International Conference on Image Processing*, pp. 889–892, Sept. 2010.

[15] S. C. Park, M. K. Park, and M. G. Kang, "Super-resolution image reconstruction: a technical overview," *IEEE Signal Processing Magazine*, vol. 20, pp. 21–36, May 2003.

[16] M. Bierling, "Displacement estimation by hierarchical block matching," in *Visual Communications and Image Processing*, 1988.

[17] R. Tsai and T. Huang, "Multiple frame image restoration and registration," *In Advances in Computer Vision and Image Processing*, pp. 317–339, 1984.

[18] P. Milanfar, *Super-Resolution Imaging*. CRC Press, 2011.

[19] M. Elad and D. Datsenko, "Example-based regularization deployed to super-resolution reconstruction of a single image," *The Computer Journal*, vol. 52, no. 1, pp. 15–30, 2009.

[20] S. Kim, N. Bose, and H. Valenzuela, "Recursive reconstruction of high resolution image from noisy undersampled multiframes," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 38, pp. 1013–1027, June 1990.

[21] N. Bose, H. Kim, and H. Valenzuela, "Recursive implementation of total least squares algorithm for image reconstruction from noisy, undersampled multiframes," in *ICASSP-93. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, pp. 269–272 vol.5, Apr. 1993.

[22] W. Yu-Su and S. P. Kim, "High-resolution restoration of dynamic image sequences," *International Journal of Imaging Systems and Technology*, vol. 5, no. 4, pp. 330–339, 1994.

[23] H. Ur and D. Gross, "Improved resolution from subpixel shifted pictures," *CVGIP: Graph. Models Image Process.*, vol. 54, pp. 181–186, Mar. 1992.

[24] A. Papoulis, "Generalized sampling theorem," in *IEEE Trans. Circuits Syst.*, vol. 24, pp. 652–654, Nov. 1977.

[25] J. Brown, J., "Multi-channel sampling of low-pass signals," *IEEE Transactions on Circuits and Systems*, vol. 28, pp. 101–106, Feb. 1981.

[26] M. Alam, J. Bognar, R. Hardie, and B. Yasuda, "Infrared image registration and high-resolution reconstruction using multiple translationally shifted aliased video frames," *IEEE Transactions on Instrumentation and Measurement*, vol. 49, pp. 915–923, Oct. 2000.

[27] N. Nguyen and P. Milanfar, "An efficient wavelet-based algorithm for image super-resolution," in *Proceedings of International Conference on Image Processing*, vol. 2, pp. 351–354 vol.2, Sept. 2000.

[28] A. Danielyan, R. Foi, V. Katkovnik, and K. Egiazarian, "Image and video super-resolution via spatially adaptive blockmatching filtering," in *Proceedings of International Workshop on Local and Non-Local Approximation in Image Processing (LNLA)*, 2008.

[29] A. Danielyan, R. Foi, V. Katkovnik, and K. Egiazarian, "Image upsampling via spatially adaptive block-matching filtering," in *Proc. of 16th European Signal Processing Conference, EUSIPCO2008*, 2008.

[30] R. Hardie, "A fast image super-resolution algorithm using an adaptive wiener filter," *IEEE Transactions on Image Processing*, vol. 16, pp. 2953–2964, Dec. 2007.

[31] M. Protter, M. Elad, H. Takeda, and P. Milanfar, "Generalizing the nonlocal-means to super-resolution reconstruction," *IEEE Transactions on Image Processing*, vol. 18, pp. 36–51, Jan. 2009.

[32] H. Takeda, P. Milanfar, M. Protter, and M. Elad, "Super-resolution without explicit subpixel motion estimation," *IEEE Transactions on Image Processing*, vol. 18, pp. 1958–1975, Sept. 2009.

[33] D. Capel, *Image Mosaicing and Super-resolution*. Springer, 2004.

[34] Y. Weiss and W. T. Freeman, "What makes a good model of natural images?," *CVPR '07. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[35] R. R. Schultz and R. L. Stevenson, "Extraction of high-resolution frames from video sequences," *IEEE Transactions on Image Processing*, vol. 5, pp. 996–1011, 1996.

[36] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D*, vol. 60, pp. 259–268, Nov. 1992.

[37] Y. Li and F. Santosa, "A computational algorithm for minimizing total variation in image restoration," *IEEE Trans. Image Processing*, vol. 5, pp. 987–995, 1996.

[38] T. F. Chan, S. Osher, and J. Shen, "The digital tv filter and nonlinear denoising," *IEEE Trans. Image Process*, vol. 10, pp. 231–241, 2001.

[39] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar, "Fast and robust multi-frame super-resolution," *IEEE Transactions on Image Processing*, vol. 13, pp. 1327–1344, 2003.

[40] B. C. Tom and A. K. Katsaggelos, "Reconstruction of a high-resolution image by simultaneous registration, restoration, and interpolation of low-resolution images," in *Proceedings of the IEEE International Conference on Image Processing*, pp. 539–542, 1995.

[41] N. A. Woods, N. P. Galatsanos, S. Member, and A. K. Katsaggelos, "Stochastic methods for joint registration, restoration, and interpolation of multiple undersampled images," *IEEE Trans. Image Process*, vol. 15, pp. 201–213, 2006.

[42] E. Kaltenbacher and R. Hardie, "High resolution infrared image reconstruction using multiple, low resolution, aliased frames," in *NAECON 1996. Proceedings of the IEEE 1996 National Aerospace and Electronics Conference*, vol. 2, pp. 702–709 vol.2, May 1996.

[43] A. Patti, M. Sezan, and A. Tekalp, "Robust methods for high-quality stills from interlaced video in the presence of dominant motion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 328–342, Apr. 1997.

[44] A. Patti, M. Ibrahim Sezan, and A. Murat Tekalp, "High-resolution image reconstruction from a low-resolution image sequence in the presence of time-varying motion blur," in *ICIP-94., IEEE International Conference on Image Processing*, vol. 1, pp. 343–347 vol.1, Nov. 1994.

[45] A. Patti, M. Sezan, and A. Murat Tekalp, "Superresolution video reconstruction with arbitrary sampling lattices and nonzero aperture time," *IEEE Transactions on Image Processing*, vol. 6, pp. 1064–1076, Aug. 1997.

[46] P. Eren, M. Sezan, and A. Tekalp, "Robust, object-based high-resolution image reconstruction from low-resolution video," *IEEE Transactions on Image Processing*, vol. 6, pp. 1446–1451, Oct. 1997.

[47] C. Wang, G. Yang, and Y.-P. Tan, "Reconstructing videos from multiple compressed copies," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, pp. 1342–1351, Sept. 2009.

[48] M. Sezan, "An overview of convex projections theory and its application to image recovery problems," *Ultramicroscopy*, vol. 40, no. 1, pp. 55–67, 1992.

[49] D. Mudugamuwa, X. He, D. Wei, and C.-H. Ahn, "Super-resolution by prediction based sub-pel motion estimation," in *IVCNZ '09. 24th International Conference on Image and Vision Computing*, pp. 282–287, Nov. 2009.

[50] N. Paragios, Y. Chen, and O. Faugeras, *Handbook of mathematical models in computer vision*. Springer, 2006.

[51] A. Jain, *Fundamentals of Digital Image Processing*, pp. 320–322. Prentice-Hall, 1986.

[52] O. Lee and Y. Wang, "Motion-compensated prediction using nodal-based deformable block matching," *Journal of Visual Communication and Image Representation*, vol. 6, no. 1, pp. 26–34, 1995.

[53] G. Al-Regib, Y. Altunbasak, and R. Mersereau, "Hierarchical motion estimation with content-based meshes," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 1000–1005, Oct. 2003.

[54] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Transactions on Image Processing*, vol. 16, pp. 349–366, Feb. 2007.

[55] G. Callico, S. Lopez, O. Sosa, J. Lopez, and R. Sarmiento, "Analysis of fast block matching motion estimation algorithms for video super-resolution systems," *IEEE Transactions on Consumer Electronics*, vol. 54, pp. 1430–1438, Aug. 2008.

[56] J. Weickert, A. Bruhn, T. Brox, and N. Papenberg, "A survey on variational optic flow methods for small displacements," in *Mathematical Models for Registration and Applications to Medical Imaging* (H.-G. Bock, F. Hoog, A. Friedman, A. Gupta, H. Neunzert, W. R. Pulleyblank, T. Rusten, F. Santosa, A.-K. Tornberg, V. Capasso, R. Mattheij, H. Neunzert, O. Scherzer, and O. Scherzer, eds.), vol. 10 of *Mathematics in Industry*, pp. 103–136, Springer Berlin Heidelberg, 2006.

[57] M. Chan, Y. Yu, and A. Constantinides, "Variable size block matching motion compensation with applications to video coding," *IEE Proceedings I on Communications, Speech and Vision*, vol. 137, pp. 205–212, Aug. 1990.

[58] CCITT, "Codec for audiovisual services at n x 384 kbits/s," *Fascicle III.5, Rec. H.261*, 1988.

[59] G. de Haan, *Video processing for multimedia systems*. Eindhoven, 2000.

[60] I. Richardson, *H.264 and MPEG-4 video compression: video coding for next-generation multimedia*. Wiley, 2003.

[61] G. Giunta, "Fine estimators of two-dimensional parameters and application to spatial shift estimation," *IEEE Transactions on Signal Processing*, vol. 47, pp. 3201–3207, Dec. 1999.

[62] P. Hill, T. Chiew, D. Bull, and C. Canagarajah, "Interpolation free subpixel accuracy motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, pp. 1519–1526, Dec. 2006.

[63] K. Panusopone and D. Baylon, "An analysis and efficient implementation of half-pel motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, pp. 724–729, Aug. 2002.

[64] S. Chan, D. Vo and, and T. Nguyen, "Subpixel motion estimation without interpolation," in *2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pp. 722–725, Mar. 2010.

[65] H. B. Yin, X. Z. Fang, H. Yang, S. Y. Yu, and X. K. Yang, "Motion vector smoothing for true motion estimation," in *ICASSP 2006. International Conference on Acoustics, Speech and Signal Processing*, vol. 2, p. II, May 2006.

[66] C. Bartels and G. de Haan, "Smoothness constraints in recursive search motion estimation for picture rate conversion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, pp. 1310–1319, Oct. 2010.

[67] S.-C. Tai, Y.-R. Chen, Z.-B. Huang, and C.-C. Wang, "A multi-pass true motion estimation scheme with motion vector propagation for frame rate up-conversion applications," *Journal of Display Technology*, vol. 4, pp. 188–197, June 2008.

[68] J. Huska and P. Kulla, "A new recursive search with multi stage approach for fast block based true motion estimation," in *17th International Conference, Radioelektronika, 2007.*, pp. 1–6, Apr. 2007.

[69] Y.-K. Chen, Y.-T. Lin, and S. Kung, "A feature tracking algorithm using neighborhood relaxation with multi-candidate pre-screening," in *International Conference on Image Processing, 1996*, vol. 1, pp. 513–516 vol.2, Sept. 1996.

[70] G. de Haan, P. Biezen, H. Huijgen, and O. Ojo, "True-motion estimation with 3-d recursive search block matching," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, pp. 368–379, 388, Oct. 1993.

[71] J. Konrad and E. Dubois, "Bayesian estimation of motion vector fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 910–927, Sept. 1992.

[72] B. Mccane, B. Galvin, and K. Novins, "On the evaluation of optical flow algorithms," in *Fifth International Conference on Control, Automation, Robotics and Vision, Singapore*, pp. 1563–1567, 1998.

[73] M. Werlberger, T. Pock, and H. Bischof, "Motion estimation with non-local total variation regularization," in *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2464 –2471, June 2010.

[74] L. Xu, J. Jia, and Y. Matsushita, "Motion detail preserving optical flow estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, p. 1, 2011.

[75] D. Tzovaras, M. G. Strintzis, and H. Sahinoglou, "Evaluation of multiresolution block matching techniques for motion and disparity estimation," *Signal Processing: Image Communication*, vol. 6, no. 1, pp. 59 – 67, 1994.

[76] M. Liu and Y. Shen, "Multiframe super resolution based on block motion vector processing and kernel constrained convex set projection," vol. 7257, p. 72571J, SPIE, 2009.

[77] H. Tamura, S. Mori, and T. Yamawaki, "Textural features corresponding to visual perception," *IEEE Transactions on Systems, Man and Cybernetics.*, vol. 8, pp. 460 –473, June 1978.

[78] E. Francois, J.-F. Vial, and B. Chupeau, "Coding algorithm with region-based motion compensation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 97 –108, Feb. 1997.

[79] C. W. Therrien, *Decision estimation and classification: an introduction to pattern recognition and related topics*. New York, NY, USA: John Wiley & Sons, Inc., 1989.

[80] L. Vandendorpe, L. Cuvilier, B. Maison, P. Queluz, and P. Delogne, "Motion-compensated conversion from interlaced to progressive formats," *Signal Processing: Image Communication*, vol. 6, no. 3, pp. 193 – 211, 1994.

[81] Graham and Thomas, "A comparison of motion-compensated interlace-to-progressive conversion methods," *Signal Processing: Image Communication*, vol. 12, no. 3, pp. 209 – 229, 1998.

[82] A. Bovik, *The essential guide to video processing*. Academic Press/Elsevier, 2009.

[83] S.-C. Han and J. Woods, "Adaptive coding of moving objects for very low bit rates," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 56–70, Jan. 1998.

[84] Y.-L. Chan and W.-C. Siu, "Reliable block motion estimation through the confidence measure of error surface," *Signal Process.*, vol. 76, pp. 135–146, July 1999.

[85] L. Hill and T. Vlachos, "Fast motion estimation using reliability weighted robust search," *Electronics Letters*, vol. 37, pp. 418–420, Mar. 2001.

[86] E. Francois, J.-F. Vial, and B. Chupeau, "Coding algorithm with region-based motion compensation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 97–108, Feb. 1997.

[87] N. Oboukhova and B. Timofeev, "Calculation and application optical flow (motion vectors) with the preliminary estimation of their reliability," in *ISCE '06. IEEE Tenth International Symposium on Consumer Electronics*, pp. 1–4, 2006.

[88] K.-S. Choi and S.-J. Ko, "Hierarchical motion estimation algorithm using reliable motion adoption," *Electronics Letters*, vol. 46, no. 12, pp. 835–837, 2010.

[89] D. Wang, A. Vincent, and P. Blanchfield, "Hybrid de-interlacing algorithm based on motion vector reliability," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, pp. 1019–1025, Aug. 2005.

[90] W. Zhao and N. Nandhakumar, "Effects of camera alignment errors on stereoscopic depth estimates," *Pattern Recognition*, vol. 29, no. 12, pp. 2115 – 2126, 1996.

[91] T. Dang, C. Hoffmann, and C. Stiller, "Continuous stereo self-calibration by camera parameter tracking," *IEEE Transactions on Image Processing*, vol. 18, pp. 1536 – 1550, July 2009.

[92] I. E. Pekkucuksen, A. U. Batur, and B. Zhang, "A real-time misalignment correction algorithm for stereoscopic 3d cameras," SPIE, 2012.

[93] F. Zilly, P. Eisert, and P. Kauff, "Real-time analysis and correction of stereoscopic hdtv sequences," CVMP, 2009.

[94] E. S. Mcvey and J. W. Lee, "Some accuracy and resolution aspects of computer vision distance measurements," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-4, pp. 646 –649, Nov. 1982.

[95] M. Santoro, G. AlRegib, and Y. Altunbasak, "Motion estimation using block overlap minimization," in *Submitted to International Workshop on Multimedia Signal Processing (MMSP)*, 2012.

[96] A. Criminisi, I. Reid, and A. Zisserman, "A plane measuring device," *Image and Vision Computing*, vol. 17, no. 8, pp. 625 – 634, 1999.

[97] S. Farsiu, M. Elad, and P. Milanfar, "Constrained, globally optimal, multi-frame motion estimation," in *IEEE Workshop on Statistical Signal Processing*, pp. 1396–1401, 2005.

[98] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1330 – 1334, Nov. 2000.

[99] J. Y. Bouguet, "Camera calibration toolbox for Matlab," 2008.